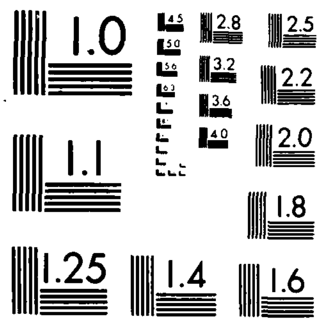
AD-A173 957 THE COMBUSTION DIAGNOSTICS LABORATORY DATA ACQUISITION AND CONTROL SYSTEM(U) ARMY BALLISTIC RESEARCH LAB ABERDEEN PROVING GROUND MD M A DEWILDE ET AL. SEP 86 1/1

UNCLASSIFIED BRL-TR-2758 F/G 9/2 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(12)

AD

US ARMY
MATERIEL
COMMAND

TECHNICAL REPORT BRL-TR-2758

AD-A173 957

THE COMBUSTION DIAGNOSTICS LABORATORY DATA ACQUISITION AND CONTROL SYSTEM

Mark A. DeWilde
Calvin E. Weaver

DTIC
ELECTE
NOV 1 0 1986
S D
D

September 1986

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

US ARMY BALLISTIC RESEARCH LABORATORY
ABERDEEN PROVING GROUND, MARYLAND

DTIC FILE COPY

86 11 1

Destroy this report when it is no longer needed.
Do not return it to the originator.

Additional copies of this report may be obtained from the National Technical Information Service, U. S. Department of Commerce, Springfield, Virginia 22161.

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

The use of trade names or manufacturers' names in this report does not constitute indorsement of any commercial product.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Technical Report BRL-TR-2758	2. GOVT ACCESSION NO. A173957	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) THE COMBUSTION DIAGNOSTICS LABORATORY DATA ACQUISITION AND CONTROL SYSTEM		5. TYPE OF REPORT & PERIOD COVERED Final 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Mark A. DeWilde Calvin E. Weaver		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS US Army Ballistic Research Laboratory ATTN: SLCBR-IB Aberdeen Proving Ground, MD 21005-5066		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 1L161102AH43
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Ballistic Research Laboratory ATTN: SLCBR-DD-T Aberdeen Proving Ground, MD 21005-5066		12. REPORT DATE September 1986 13. NUMBER OF PAGES 66
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION DOWNGRADING SCHEDULE NA

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; Distribution Unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Laboratory automation
Laboratory computers
DEC PDP-11 computer
Instrument interfacing
A/D conversion

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

meg/jmk

A versatile laboratory data acquisition and control computer that was designed and built for automation of a laser spectroscopy combustion diagnostics laboratory is described. The general purpose interface hardware and software is discussed in detail, along with general architectural philosophy and programming methodology. The architecture provided maximum adaptability to a wide range of experimental situations, while retaining user friendliness. The special software interfaces developed for this system

DD FORM 1473 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. Abstract (Cont'd).

allowed the use of high level languages for most tasks.

ACKNOWLEDGEMENTS

The authors would like to thank the BRL machine shop for fabricating the various patch panels in the system.

Accesion For		
NTIS CRA&I	☑	
DTIC TAB	☐	
Unannounced	☐	
Justification		
By		
Distribution/		
Availability Codes		
Dist	Avail and/or Special	
A-1		

QUALITY INSPECTED 4

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

A combustion diagnostics effort that this work supported was aimed at the application of laser-based probes to the study of flame chemistry. These probes include Raman spectroscopy, laser induced fluorescence, multiphoton laser induced fluorescence/ionization spectroscopy, and others. These techniques require numerous types of control signals and adjustments, and produce large quantities of data requiring further numerical processing for interpretation. As a typical example, a spontaneous Raman spectroscopy setup requires a laser beam shutter to be opened and closed, a spectrometer to be scanned, a burner that provides the flame under study to be moved so that different parts of the flame may be probed, and an optical multichannel analyzer that senses the signal light be read into digital form. In a simple profiling of a flame, it is possible that well over 1000 individual settings and operations be accomplished by an operator in order to take such data. If an automatic system were not utilized, such measurements would require a massive quantity of human intervention and would be prohibitively expensive. In such a situation, the purpose of automation is that of cost savings and release from tedium. In other cases, however, bringing the power of computer automation to bear on a problem makes possible measurements that are beyond the capabilities of non-computerized instruments and humans to accomplish. In addition, many modern state-of-the-art instruments are not designed to perform their full range of capabilities without the aid of a computer. In both of the latter cases, the decision to fully automate is unavoidable. It was a result of all of these reasons that the systems that are described in this report were developed. Much of the hardware is commercially available, and that which is not will be described. It is not the intention of this report to document all of the instrumentation that has been connected to these systems, but rather to lay the foundations of the general purpose portions of the systems that are built upon for special purposes. Some of these special purpose systems will be described in later reports by the authors, and will reference this document.

II. THE ARCHITECTURE OF THE MAIN CONTROL COMPUTERS AND SYSTEM OVERVIEW

Unless otherwise stated, all model numbers given will refer to those of the Digital Equipment Corporation. This is not an endorsement by the US Government of these products, but merely a convenient means to provide references to more extensive specifications for these instruments (see References 1 through 5 for complete descriptions of these items). All of the systems (four at the time of writing) are based on the model PDP-11 computer. This machine is a general purpose 16-bit mini/micro computer, that has gained widespread use so as to have a large base of users and available software packages. Three of the many available models of this machine are currently in use, Model 11/34, Model 11/04, and Model 11/03. The primary difference between the first and the latter two is speed, and available memory space. In the 11/34, 256 kilobytes are available, whereas in the latter two, only 64 kilobytes are available. The primary difference between the 11/34, 11/04, and the 11/03 is that the 11/03 is really a microcomputer with a different internal bus structure (Q-bus) than have the minicomputers 11/34 and 11/04 (Unibus). The system architectures are shown in Figures 1 and 2. A "base" configuration (Figure 1) consists of the central processing unit (CPU), 64 Kbytes of memory, 2 serial line units (DL-11) providing RS-232C data

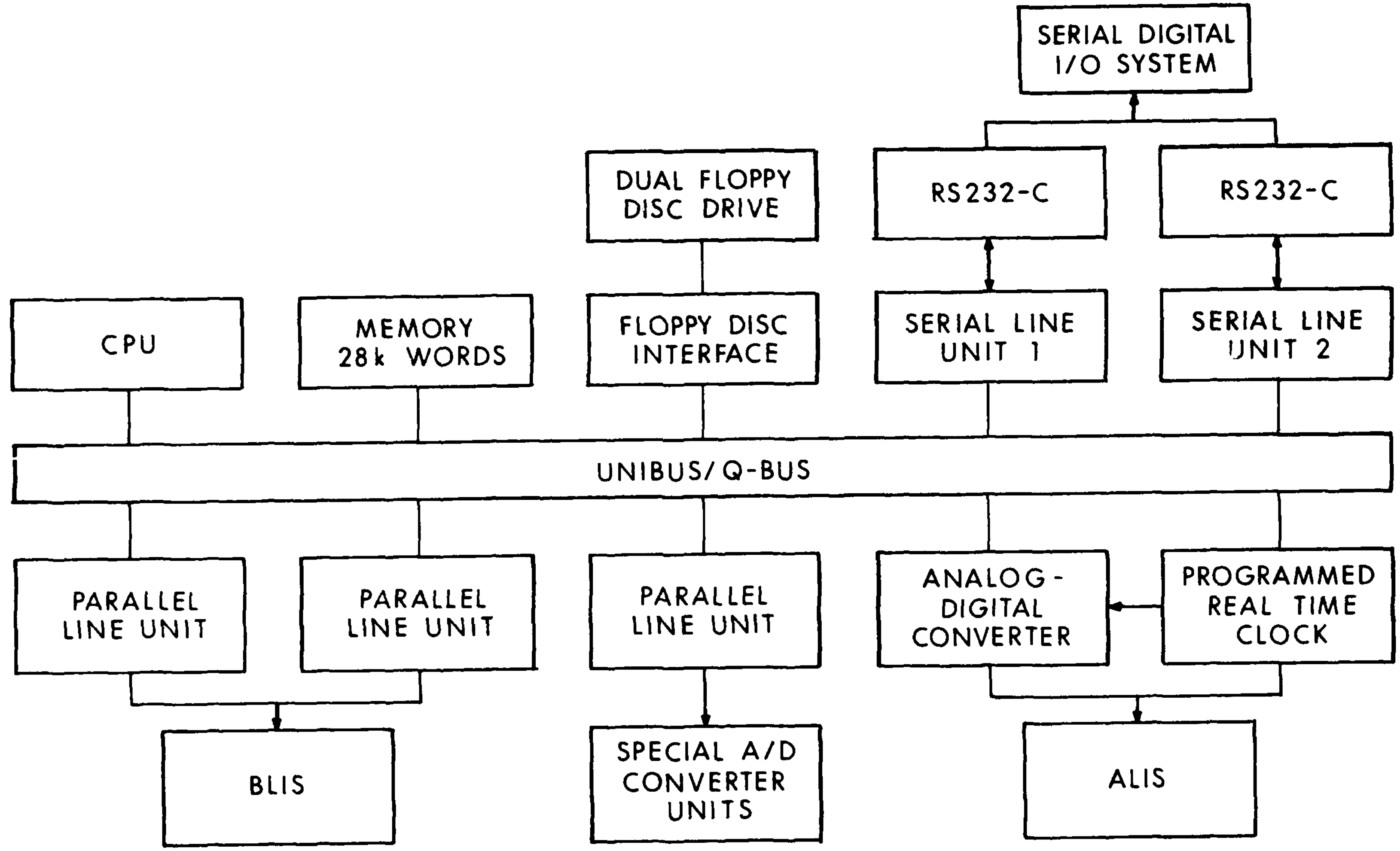


Figure 1. System Base Configuration

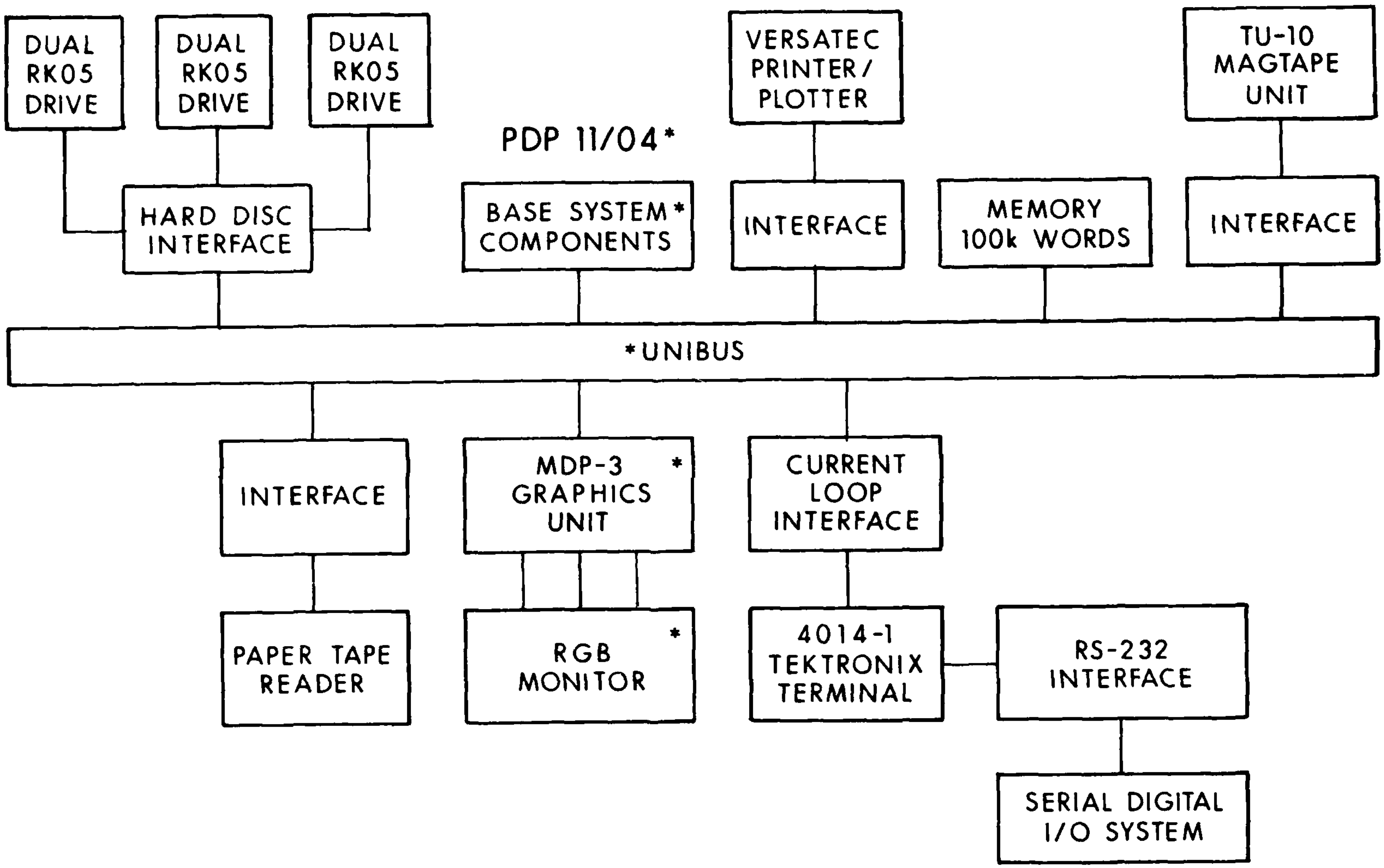


Figure 2. Actual System Configuration, 11/34 And 11/04

communications to the system terminal, a 1 Megabyte floppy disc system (RX02 type), 3 parallel line I/O units (type DR-11C), and an analog to digital converter system (AD-11C). Included as a part of the A/D system is also a programmable real-time clock, model KW-11P or KWV-11C for the Q-bus systems. Two of the parallel line units are dedicated to an input/output system of line driver-receivers that provide optically isolated TTL (Transistor-Transistor Logic) binary control lines. This Binary Laboratory Interface System (BLIS) provides the basis for interfacing the laboratory instruments to the system. This system will be discussed later in this document. The analog to digital converter system has a digital resolution of 12 bits, and a conversion time of 25 microseconds. The third of the above mentioned parallel line units is dedicated to higher speed analog to digital conversion systems that are designed and built in-house, and are outside the scope of this document.[6 7]

Additions to the base architecture were made on the various machines as shown in Figure 2. The Unibus machines (11/34 and 11/04) are equipped with raster-scan display driver devices that connect to video monitors. These are type MDP-3 from Computer Design and Applications, Inc. They are 256 x 256 raster size devices, made to drive RGB color monitors, although at the time of writing, black and white monitors were actually in use. These found use in data display of graphic nature and do not therefore require that a graphics computer terminal be connected to the system for graphic output. The system 11/34 is also provided with 800 BPI magnetic tape, paper tape reader, Versatec raster-type electrostatic printer-plotter (Model D1200), and a Tektronix 4014-1 high resolution graphics terminal with the capability of screen dump to the Versatec for hardcopy. This terminal also was capable of being connected to the BRLNET by switch for connection to any of the BRL computers, a capability used for local graphics, and making local copies of those graphics. Finally, the 11/34 is also provided with hard disc drives providing the functional equivalent of six DEC RK-05 type drives. Each RK unit provides 2.5 Megabytes of storage, and is considerably faster than the floppy disc drives on the systems. This set of differences built on top of the base systems gives some of the systems special capabilities over the others for special purposes, and was done since not all laboratory setups required all of the capabilities of the largest (11/34) system, and could be done with less expensive configurations.

The description up to now has centered on hardware, so that a discussion of software is in order. The operating system in use is DEC's RT-11 single job, single user real time system. This system was chosen for the reasons of simplicity, small memory requirements (~8 Kbytes), and the fact that all system processes may be stopped for servicing of real-time events, without extensive systems programming knowledge. The system supports plain English language commands, provides on-line help to users, and has abundant utilities for this environment. The languages installed on this system are FORTRAN-IV, Pascal, and MACRO-11, the machine's assembly language. By adhering to certain calling and return conventions, all of these languages may be used as parts of a single program, so that the strengths of each for particular types of tasks may be used without sacrificing those of the others. It is not uncommon for a FORTRAN program to call a MACRO subroutine, or to have FORTRAN and MACRO procedures called from a Pascal program. The transparency to the user is provided by the way in which the operating system builds executable modules. The first step is to use one of the provided editor programs on the system to create a source file. The provided editors are called "KED", which is the

video terminal keypad editor and resembles the "EDT" editor under DECs VAX/VMS operating system. The two others are line editors for any terminal, and are called "TECO" and "EDIT". These use single letter mnemonics for commands, and control characters as delimiters for text and commands.

After the source code files are created, compilers are invoked to verify that there have been no grammatical errors, and to produce so called "object" files. These object files all have the same internal format and are no longer tied to the language of the source file that they represent. The FORTRAN compiler is the standard one suplied by DEC, but the Pascal compiler was that of Oregon Software, Inc. This is a five pass optimizing compiler that, for the same functional program, will produce considerably smaller and faster code than that produced by the non-optimized FORTRAN compiler. The final stage in the production of the executable code is done by the system's "LINK" program. This program has the task of resolving all inter-module references, calls to system library functions, relocation and binding of all relocatable code sections, and setting up all overlay handling for the overlay feature. The overlay feature is the way that the system compensates for the fact that only about 48 Kbytes of memory is available for any user program. In practice, the usable memory is broken into two parts: a root region and an overlay region. All code in the root region is resident the entire time the program is in execution. The overlay region area is used to load in "overlays", multiple modules of code that are needed only one at a time. When the program is being built, the writer structures it into as many subroutines or procedures as is reasonable, and places each of these into its own module. These modules are then grouped into "segments" by appropriate commands to the linker. In the execution of the final program, when the current command in the root segment calls a subroutine or procedure that is located in an overlay segment, that segment is loaded into the overlay region, and execution continues. Any code that was in the overlay region is overwritten. Since the overlay segments of code are on disk, and only a copy of the code made in the overlay region, it is not necessary to copy the overlay segment in memory back out onto disc before copying the new overlay segment into the overlay region. There can be many overlay segments which can be loaded into the overlay region, but only one root segment. From the description of the way segments are loaded and overwritten, it becomes obvious that no routine in one overlay segment may call one in another. Even with these restrictions, clever use of this overlay feature allows large programs be written that will fit into the small space available on these machines.

Due to the fact that the user is the sole one on the system, language extensions are provided that enable the user to address absolute locations in the system memory (i.e., virtual addressing is not used - all addresses are physical). Since this machine uses memory mapped input/output, and thus treats discs, terminals, etc., as memory locations, it is possible for users to program, in high level languages such as FORTRAN and Pascal, hardware devices to do things that usually are done solely in assembly language, provided the speed requirements are not too demanding. Software handlers for the A/D converters and other types of hardware have been written in both FORTRAN and Pascal, and have been used routinely for years.

The remaining elements in the operating system provide utilities to do device maintenance functions such as checking for bad blocks, formatting, initializing, etc., to do file transfers, creates and deletes, to maintain

directories, debug programs, and a multitude of other functions. The reader is referred to Reference 8 for further details. Other utilities on the system have been both acquired from commercial vendors, and written in-house. Tektronix, Inc. provided the standard Plot-10 package, which is used for generation of graphics on the 4014 and 4014 look-alike terminals in use. Versatec provided both the Versaplot package and the Pen Plotter Emulation Package for the Versatec D1200 printer/plotter. The former package provides short, high-level calls to subroutines that create completed graphs in relatively few calls. The latter package provides the user with the standard CALCOMP pen plotter control calls. Computer Design and Applications, Inc., provided the driver package for their MDP-3 raster scan graphics processor. This package provides both FORTRAN/Pascal callable subroutines, and a micro assembly language for graphics generation. Extensive utility software was also developed in-house. Since Plot-10 is a memory hog and typically adds 10 Kbytes to any program it is used with, a set of graphics primitives was written in assembly language and provided a compact means of including graphics in large programs. This package will be described in later sections. Additional packages were written to drive the DEC A/D converters, and provide the FORTRAN/Pascal programmer with transparent calls that take voltage measurements from external instruments. The drivers for this package were written in both MACRO, and Pascal. This package will also be described in later sections. As a final note, hardware/software packages were developed to interface and drive IEEE-488 bus instruments, and high speed A/D converters that were developed in-house. These packages are described in other reports by the authors.[6,7]

III. THE SERIAL DIGITAL I/O SYSTEM

As mentioned above, each base configuration includes two DL-11 type serial line units that provide RS-232C communications. One of these is hardwired to the bus addresses (i.e., memory location addresses):

Address (Octal)	Register Function
177560	Keyboard Status
177562	Keyboard Data Buffer
177564	Printer/Screen Status
177566	Printer/Screen Buffer

A memory location or register can be looked at as a cell consisting of 16 bits:

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00

For the registers above, the functions of these bits is as follows:

Keyboard Status Register

Bit	Meaning and Operation
15 - 12	Unused.
11	Receiver active. Indicates that a character is currently in the process of being received.
10 - 08	Unused.
07	Receiver done. Indicates that there is a character in the input buffer ready to be read by the program.
06	Receiver interrupt enable. Allows the receiver to interrupt the current program execution, and a handler routine to be run. See References 1 and 3 for more information on interrupts.
05 - 00	Unused

Keyboard Buffer Register

Bit	Meaning and Operation
15	Error. This bit is set whenever bits 12 - 14 are set.
14	Overrun. This bit is set whenever a character is received before the last one received has been read by the program.
13	Framing error. This bit is set whenever a break condition is encountered, no stop bits are received, or if there is a baud rate mismatch.
12	Parity error. This bit is set if the received character parity does not match the expected parity
11 - 08	Unused.
07 - 00	Received data bits.

Transmitter Status Register

Bit	Meaning and Operation
15 - 8	Unused.
07	Transmitter ready. Set when the transmitter can accept a character for transmission.
06	Transmitter interrupt enable. When set, an interrupt will be generated whenever bit 07 is set.
05 - 03	Unused.

02	Maintenance. This bit essentially connects the output of the transmitter to the input of the receiver for test purposes.
01	Unused.
00	Break. When set, the transmitter sends a break condition.

Transmitter Data Buffer

Bit	Meaning and Operation
15 - 08	Unused.
07 - 00	Transmitted data. A byte placed in these bits will be transmitted by the interface.

For the system console terminal, these buffers are handled by the operating system, and the user need not be aware of their details, except for the fact that the operating system uses interrupts generated by these registers in the handler. This becomes important if time-critical programs are running, in that terminal generated interrupts can interfere with the critical task. This can be overcome by disabling these interrupts before the time-critical task begins. The MACRO instruction to do this would be: BIC #100, @#177560 and has the sole effect of turning off bit 6 in the receiver status register. In order for the normal system terminal operation to proceed after the time critical routine is done, it is necessary to reverse the change.

The second serial line unit is identical to the first, except that its bus addresses are 175000 to 175006. After initial system startup, it is possible via software to transfer control of the system to this unit, and back again to the original. This is used to avoid the necessity of doing hardware switching of transmission lines between terminals at various locations in the laboratory rooms. The program to do this is taken from Reference 8 and follows:

```
        .TITLE  XXXXXX
        .MCALL  .MTPS, .PRINT,.EXIT
        CSRAD = YYYYYY
        OLDVEC = WWW
        VEC = ZZZ
        SYSGEN = 372
        MTTY$ = 20000
        BITMAP = 326+<VEC/20>
        BMASK = 360/<<15.*VEC-<20*<VEC/20>>>/8.>+1>;
PROC3:  MOV     @#54,R0                 ;get the start location of the monitor
        BIT     #MTTY$,SYSGEN(R0)       ;see if multi terminal option used
        BNE     2$                      ;go to 2$ if so
        .MTPS   7                       ;else, go to priority 7 (top priority)
        BISB    #BMASK,BITMAP(R0)       ;protect new console vectors
        ADD     #304,R0                 ;R0 => console register list in rmon
        MOV     #CSR,R1                 ;R1 => new CSR/data reg. list
        CLR     @(R0)                   ;get rid of old CSR
1$:     MOV     (R1)+,(R0)+             ;move in new CSR/data register addr.
```

```
        TST     @R1                     ;done?
        BMI     1$                      ;if minus, then do another
        MOV     #OLDVEC,R0              ;R0 = present console vector
        MOV     @R1,R1                  ;R1 = New console vectors
        .REPT   4
        MOV     (R0)+,(R1)+             ;load new vectors from old
        .ENDR
        .MTPS   0                       ;back to lowest priority
        .EXIT                           ;done with program
2$:     .PRINT  #NOMT                   ;error handler code
        .EXIT
NOMT:   .ASCIZ  /?MULTI TERMINAL SYSTEM, USE SET TT CONSOLE/
        .EVEN
CSR:    .WORD   CSRAD                   ;CSR/buffer/vector list area
        .WORD   CSRAD + 2
        .WORD   CSRAD + 4
        .WORD   CSRAD + 6
        .WORD   VEC
        .END    PROC3
```

Where XXXXXX = name of transfer routine, YYYYYY = 177560 ,ZZZ = 60, and WWW=140 for transfer to the serial line unit with addresses starting at 177560, and YYYYYY = 175000, ZZ = 140, WWW = 60 for transfer to the serial line unit with addresses starting at 175000. In our installation, the former set of addresses are installed in the program and the routine named "TTMAIN". The latter set are installed and the program named "TT4800" after the baud rate that that unit is set for. The programs are edited to insert changes, then assembled with the macro assembler, and linked with the linker. When either is run, control is transferred to that terminal.

The above program is useful when the multi-terminal feature of RT-11 has not been implemented in the system. In such a case, if the user wishes to drive the serial port not dedicated to the system terminal, either he must write his own handler in one of the available languages, or buy a handler from one of the software houses that sell them. The former has been used in this work, in that few of the instruments used in this laboratory actually make use of serial communications. When such communications are necessary, the multi-terminal option provided with the system and described in Reference 8 is used. As a final note about this, the graphics library described below is one such in-house written handler, made necessary by the fact that the operating system intercepts certain of the control characters necessary to transmit for graphics generation, and substitutes others.

In order to connect the eight serial line units (two on each of four machines) to any serial device in one of the seven different rooms of the laboratory, a patch panel system was devised and constructed. Figure 3 shows the schematic layout of the system, and Figures 4, 5, and 6 the drawings of the components. The jumper cables are wired as shown in Figure 6. All of the plugs and sockets (cinch DBC 25-P and DBM 25-S, respectively) are carried in the federal stock system. The FSN of the male plug is 5935-00-259-4690, and that of the female socket is 5935-00-351-6135. The housing for either is FSN 5935-00-401-6454. For cabling, a cable consisting of two shielded twisted pairs with drain wire was used, and is made by Belden Wire and Cable, stock #8723. In this work, approximately 3000 feet were used. Each laboratory plug

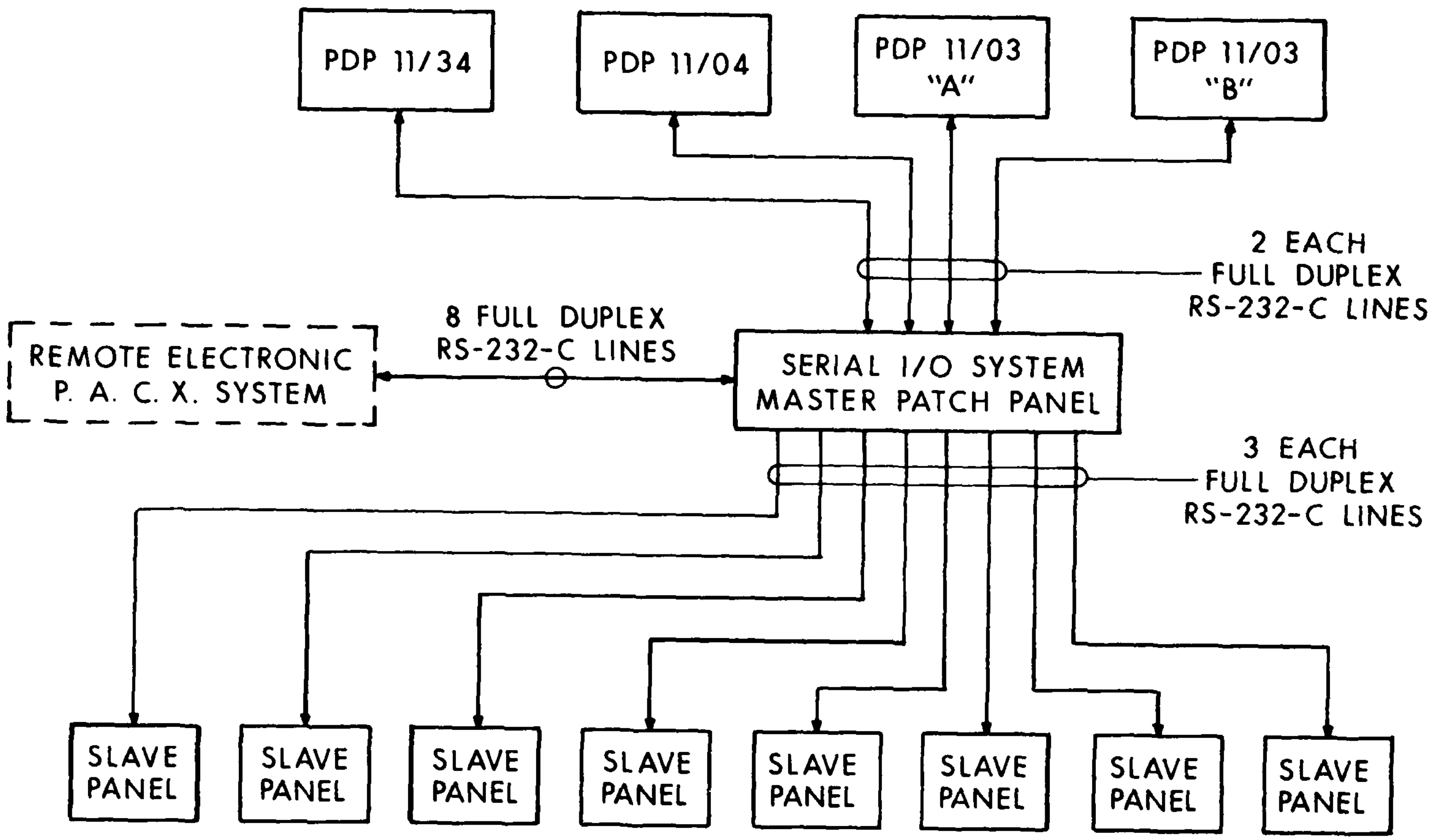


Figure 3. Serial I/O System Schematic

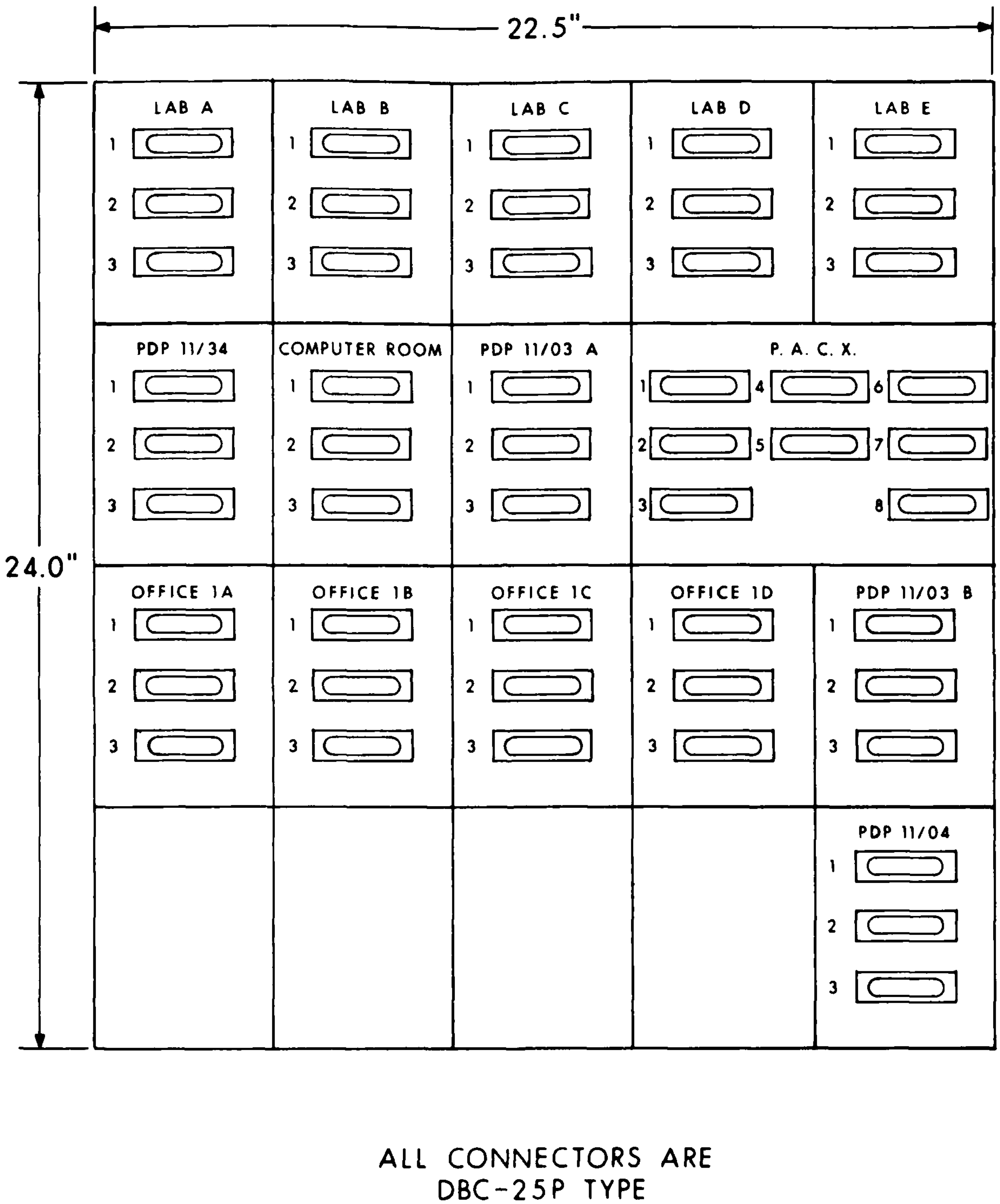
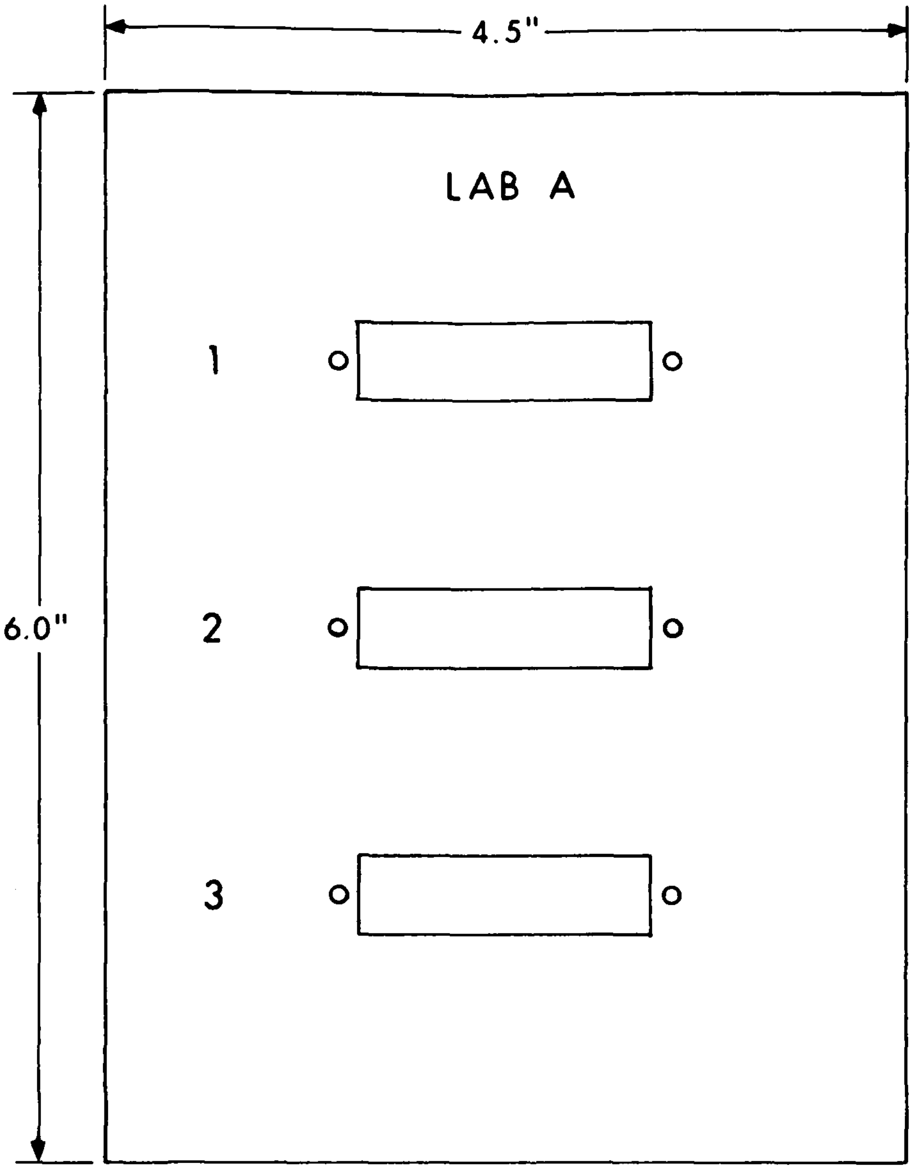


Figure 4. Serial I/O Master Patch Panel Layout



ALL HOLES FOR
DBC-25P CONNECTOR

Figure 5. Typical Slave Patch Panel

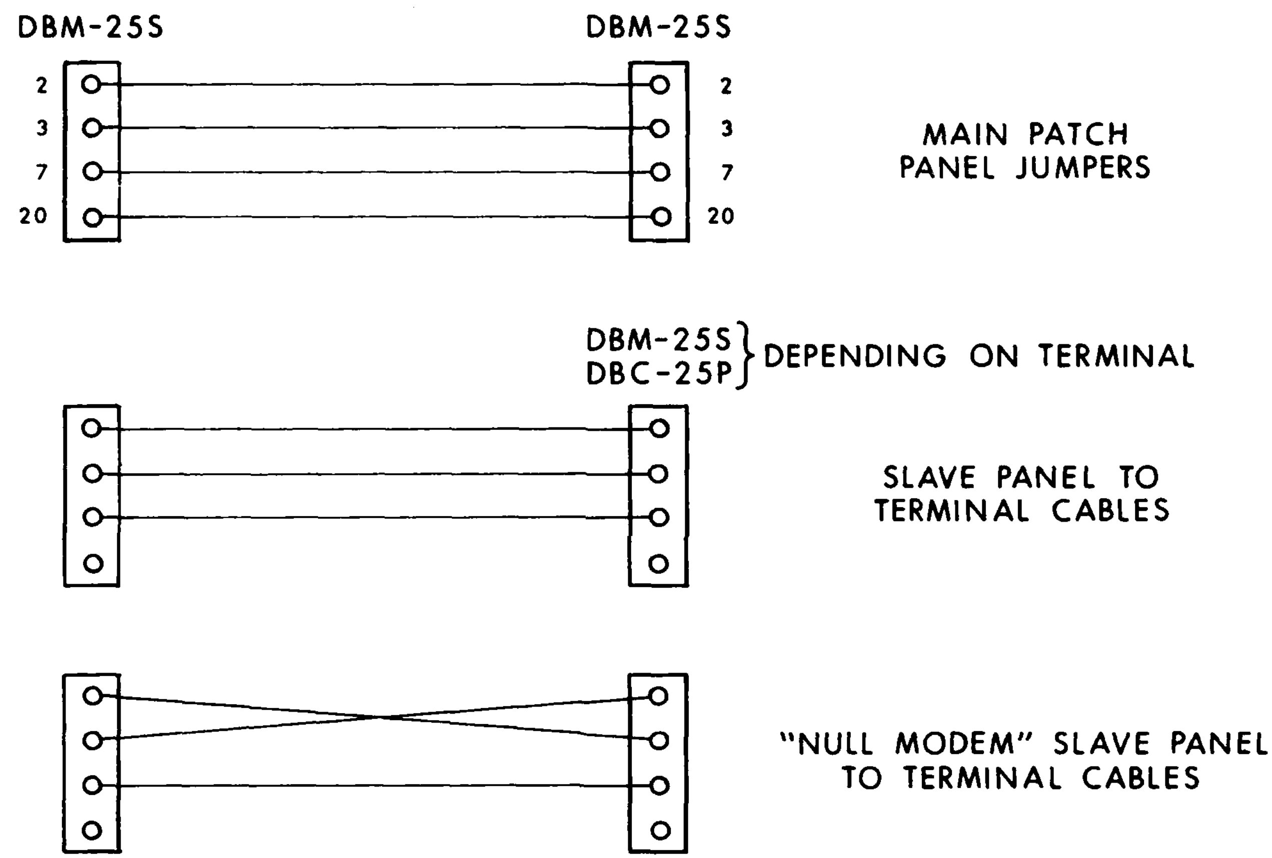


Figure 6. Jumper Schematics

panel (Figure 5) has an identical section on the master patch panel (shown in Figure 4) in the room where all four computer systems reside. By using a patch cord, one can easily connect any serial line from any of the four machines to any location in the laboratory, or offices. As a note, eight connections labeled "PACX" appear in the figures. These are lines which go elsewhere in the building to a switching network that connects to a large number of other computers, and the BRLNET. Usually one of the three plugs in each laboratory location is patched to one of these lines to give users access to the net.

IV. THE PARALLEL DIGITAL I/O SYSTEM

The acronym given to this system is BLIS, standing for the Binary Laboratory Interface System. This system was developed to control and receive signals from laboratory instruments that use parallel digital input/output lines for control and data output. Since this system was designed and put into use several years ago when the preponderance of digitally interfacable instrumentation used TTL logic, the system uses this logic standard for its input/output. As an overview, each base configuration is provided with two parallel line interfaces of the DR-11C type. These interfaces provide 16 output lines related one-to-one with the bits in an output register, 16 input lines related in the same fashion to an input register, two interrupt request input lines, two control output lines, and two status control lines. The interrupt request and control output lines are related to specific bits in the status register of the DR-11C, and the remaining two control output lines indicate to the external device that the output lines have been loaded, and that the input lines have been read. The registers and their bit assignments are as follows:

DR-11C Control and Status Register

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00

Bit	Meaning and Operation
15	Interrupt request "B" (input signal).
14 - 08	Unused.
07	Interrupt request "A" (input signal).
06	Interrupt enable "B".
05	Interrupt enable "A".
03 - 02	Unused.
01	CSR bit 1 (output signal).
00	CSR bit 0 (output signal).

Data Output Register

Bit	Meaning and Operation
15 - 00	Data out. Each bit corresponds to an output line.

Data Input Register

Bit	Meaning and Operation
15 - 00	Data input. Each bit corresponds to an input line.

The signals indicated above appear at the circuit board edge on two 44 pin Berg Electronics Company headers. The pin assignments appear in the table below. In the line of explanation of this table, the names "DROUT" refer to those signals related to the DR-11 Data output register. The signals named "DRIN" refer to those related to the bits of the DR-11 data input register. All other signals refer to those related to the bits in the command and status register. For the first set of BLIS transmitter/receivers, the DR-11C is set to the bus addresses 167750 to 167754. The second set corresponds to bus addresses 167740 to 167744. The CSR is the lowest address, the output register next, and the input register the highest address in the set of three. The outputs and inputs at the board are TTL signals. The logic levels for TTL are represented as voltage levels. A logical "1" in TTL is represented as a voltage between 3.0 and 5.0 volts. A logical "0" is represented as a voltage level between 0.0 and 0.8 volts. While it is not intuitively obvious to the non-engineer, it is virtually impossible to transmit these voltage level signals over wires of any appreciable distance without excessive electronic noise pickup, or data loss. In order to make these signals appear in laboratories that can be hundreds to thousands of feet away, a current-loop optically isolated system of transmitters/receivers was developed. A schematic for a single transmitter/receiver pair is shown in Figure 7. In order to make all of the available signals in the DR-11 interface appear in the laboratory, 18 transmit lines and 18 receive lines are needed, along with the return lines and grounding lines. Schematic diagrams of the BLIS units are shown in Figures 8-11. The cable between the receiver and transmitter boards is Belden type #8769. The circuit board substrates are from Vector Electronics, Model #3682-4. The circuit board edge connectors are 44-pin type from Amphenol Electronics, type #225-22221-101. The actual construction required that the computer end driver and receiver boards be slightly different from those at the laboratory. This was caused by several things. The first reason is that not all of the signals on the Berg headers of the DR-11C are logically grouped (i.e., both input and output signals are mixed on the same header). It was desired at the laboratory end to have all DROUT lines, CSR0 and CSR1 lines, and NDR line (which indicates that the DROUT register has been loaded) on the same output header. Similarly, it was desired that all input lines, the two interrupt request input lines and the line that indicates that the input register has been read (data transmitted) on the other header. This is a desirable situation in order to minimize the number of circuit boards needed to interface laboratory experiments. If only inputs are needed, a lab end driver and computer end receiver are needed, instead of the two boards that would be needed if the DR-11 signals were transmitted and received as they appear at the Berg headers on the interface

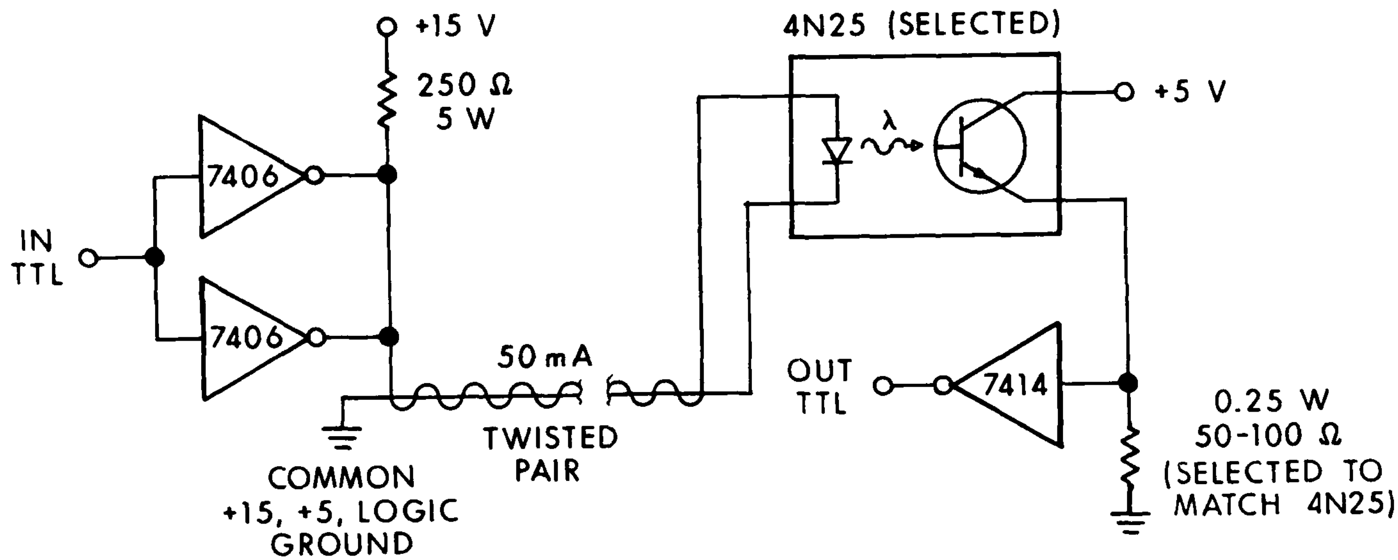


Figure 7. BLIS Transmit-Receive Base Circuit

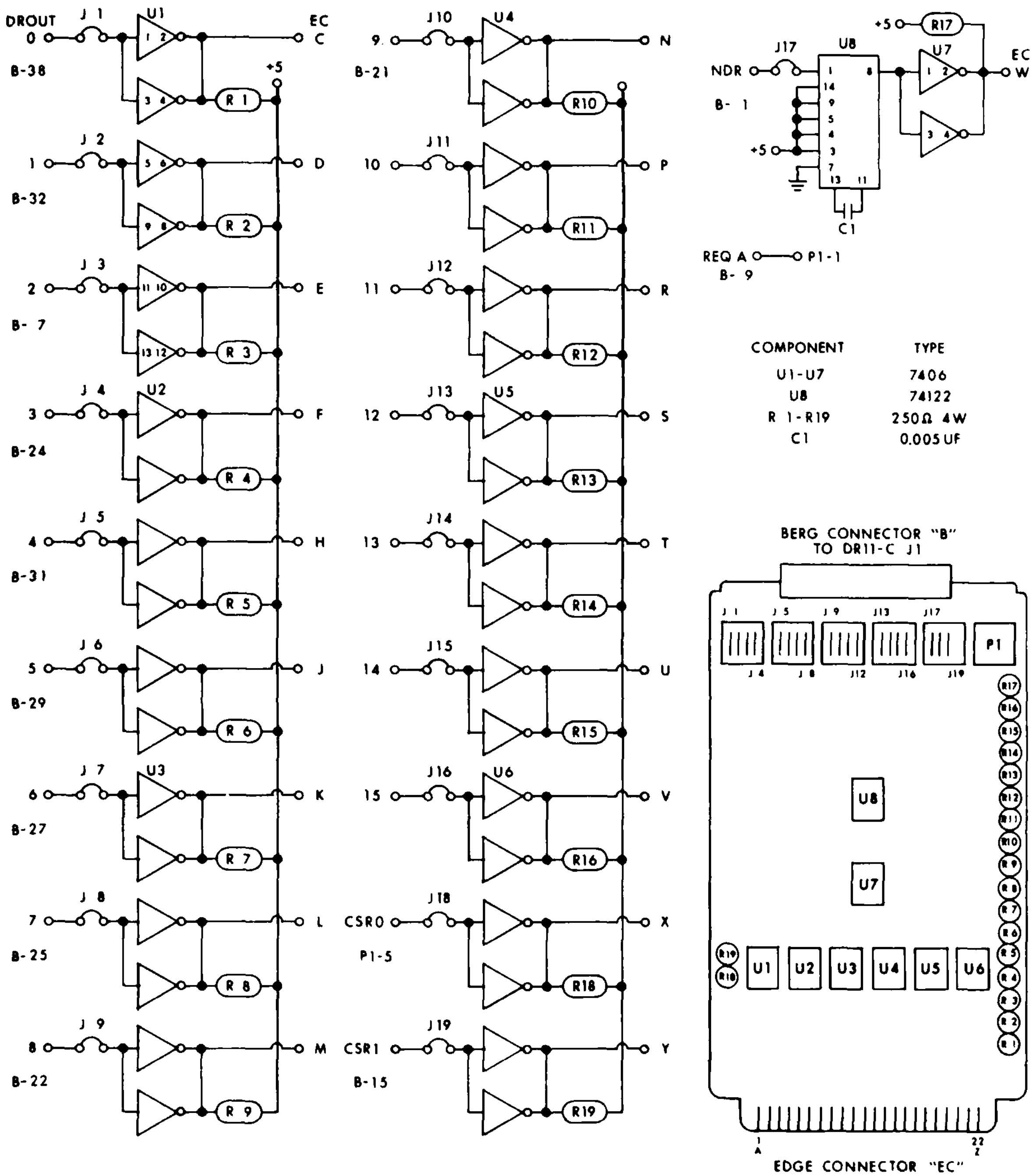
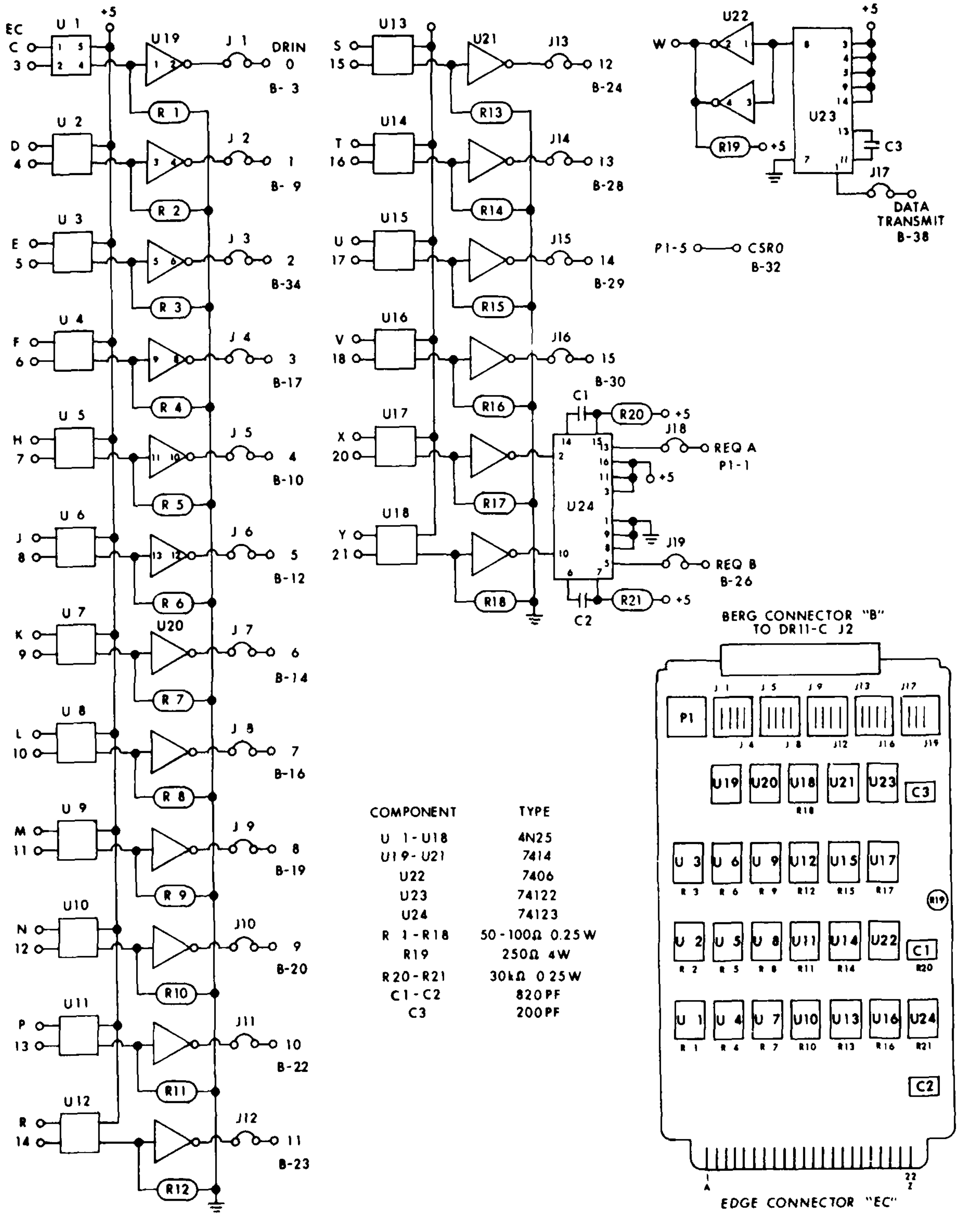


Figure 8. BLIS Computer End Driver Board



COMPONENT	TYPE
U 1-U18	4N25
U19-U21	7414
U22	7406
U23	74122
U24	74123
R 1-R18	50-100Ω 0.25W
R19	250Ω 4W
R20-R21	30kΩ 0.25W
C1-C2	820PF
C3	200PF

Figure 9. BLIS Computer End Receiver Board

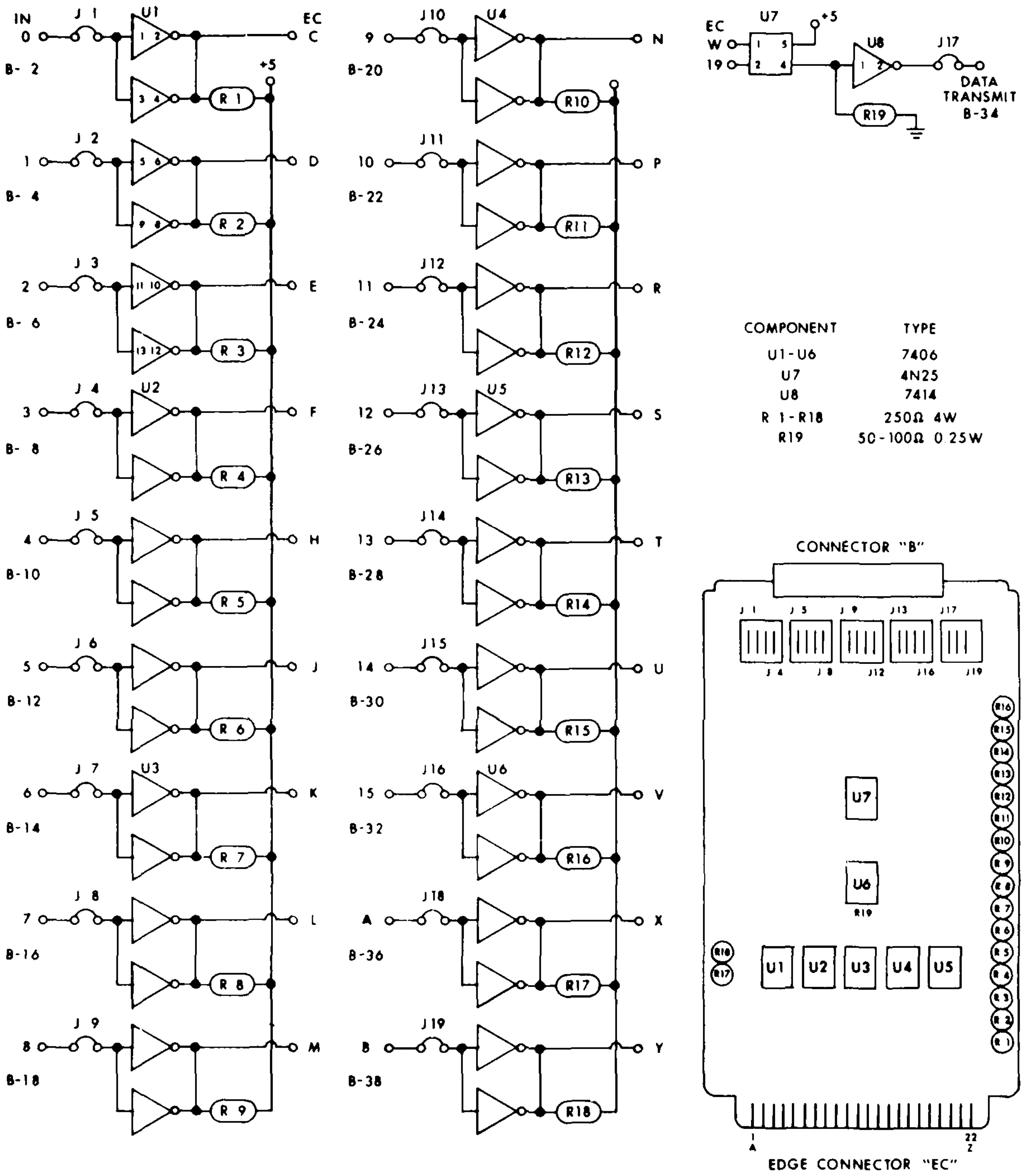


Figure 10. BLIS Laboratory End Driver Board

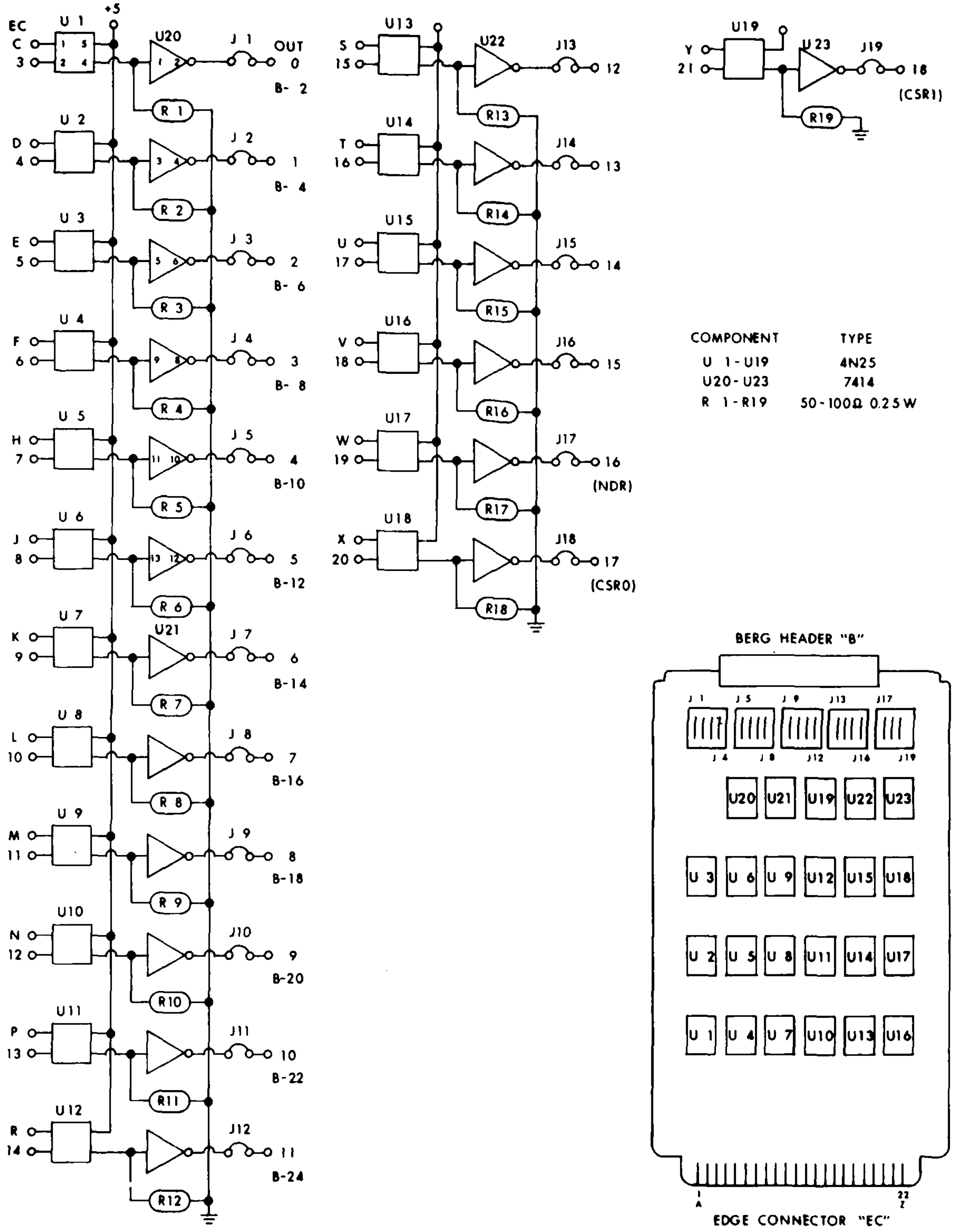


Figure 11. BLIS Laboratory End Receiver Board

board. As before, a method is needed to patch the eight sets of BLIS boards to any location in the laboratories. This is done in a way that parallels that of the serial line interface system. Figure 12 shows the master BLIS patch panel. Corresponding to each of the sections of the master patch panel there is a slave patch panel in each of the laboratories , an example of which is shown in Figure 13. The overall system schematic is shown in Figure 14. There are two power supplies at each slave panel, and two in each computer chassis to power the boards plugged into that end. One of these is a +5 volt, and the other a +15 volt. Both are switching-type supplies, and were procured from Sierracin Power Systems, Inc. The five volt was model 5A5A, and the 15 volt was Model 5A15A. The patch cables and connectors were fabricated from Vector Electronics Company's Model 838WE circuit boards, and 40 conductor ribbon cable, Alpha Wire and Cable number 3580/40. These jumpers provided a one-to-one connection between terminals in the edge connectors used, and are shown in Figure 15. The following table lists the DR-11 signals, their pin numbers on the DR board and computer end boards, the BLIS board edge finger assignments, and the laboratory end connector pin assignments.

DR-11C J1 pin	Signal Name	Board type/ Edge Finger	Board type/ Lab Berg header pin
1	New Data Ready	CD/W	LR/34
2	Ground	*	#
3	No Connection	-	-
4	Ground	*	#
5	No Connection	-	-
6	Ground	*	#
7	DROUT Bit 02	CD/E	LR/06
8	Ground	*	#
9	Interrupt request B	CR/Y (**)	LD/38
10	Ground	*	#
11	DROUT Bit 15	CD/V	LR/32
12	DROUT Bit 14	CD/U	LR/30
13	DROUT Bit 13	CD/T	LR/28
14	Ground	*	#
15	Command/Status Bit 1	CD/Y	LR/38
16	Ground	*	#
17	DROUT Bit 12	CD/S	LR/26
18	DROUT Bit 11	CD/R	LR/24
19	DROUT Bit 10	CD/P	LR/22
20	Ground	*	#
21	DROUT Bit 09	CD/N	LR/20
22	DROUT Bit 08	CD/M	LR/18
23	Ground	*	#
24	DROUT Bit 03	CD/F	LR/08
25	DROUT Bit 07	CD/L	LR/16
26	Ground	*	#
27	DROUT Bit 06	CD/K	LR/14
28	Initialize	NOT TRANSMITTED	-
29	DROUT Bit 05	CD/J	LR/12
30	Ground	*	-
31	DROUT Bit 04	CD/H	LR/10

32	DROUT Bit 01	CD/D	LR/04
33	Ground	*	#
34	New Data Rdy, Low Byte	NOT TRANSMITTED	-
35	Not Used	-	-
36	New Data Rdy,High Byte	NOT TRANSMITTED	-
37	Not Used	-	-
38	DROUT Bit 00	CD/C	LR/02
39	Not Used	-	-
40	Not Used	-	-

DR-11C J2 pin	Signal Name	Board type/ Edge Finger	Board type/ Lab Berg header pin
1	No Connection	-	-
2	Ground	*	#
3	DRIN Bit 00	CR/C,3	LD/02
4	Ground	*	#
5	Initialize	NOT TRANSMITTED	-
6	Ground	*	#
7	Initialize	NOT TRANSMITTED	-
8	Ground	*	#
9	DRIN Bit 01	CR/D,4	LD/04
10	DRIN Bit 04	CR/H,7	LD/10
11	Ground	*	#
12	DRIN Bit 05	CR/J,8	LD/12
13	No Connection	-	-
14	DRIN Bit 06	CR/K,9	LD/14
15	Ground	*	#
16	DRIN Bit 07	CR/L,10	LD/16
17	DRIN Bit 03	CR/F,6	LD/08
18	Ground	*	#
19	DRIN Bit 08	CR/M,11	LD/18
20	DRIN Bit 09	CR/N,12	LD/20
21	Ground	*	#
22	DRIN Bit 10	CR/P,13	LD/22
23	DRIN Bit 11	CR/R,14	LD/24
24	DRIN Bit 12	CR/S,15	LD/26
25	Ground	*	#
26	Interrupt Request B	CR/Y,21	LD/38
27	Ground	*	#
28	DRIN Bit 13	CR/T,16	LD/28
29	DRIN Bit 14	CR/U,17	LD/30
30	DRIN Bit 15	CR/V,18	LD/32
31	Ground	*	#
32	Command/Status Bit 0	CD/D (**)	LR/36
33	Ground	*	#
34	DRIN Bit 02	CR/E,5	LD/06
35	Not Used	-	-
36	Not Used	-	-
37	Not Used	-	-
38	Data Transmitted	CR/W	LD/34
39	Not Used	-	-
40	Not Used	-	-

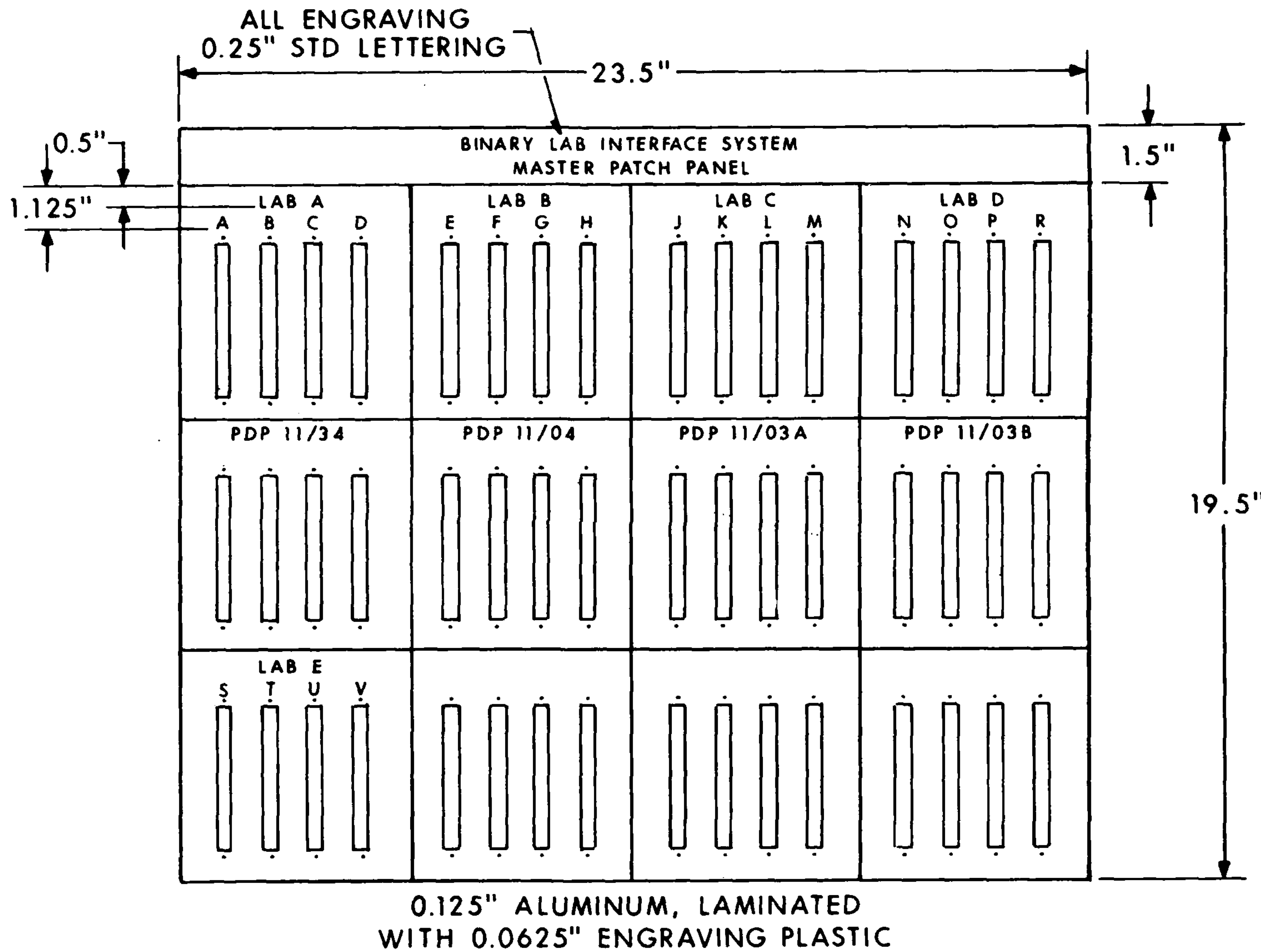


Figure 12. BLIS Master Patch Panel

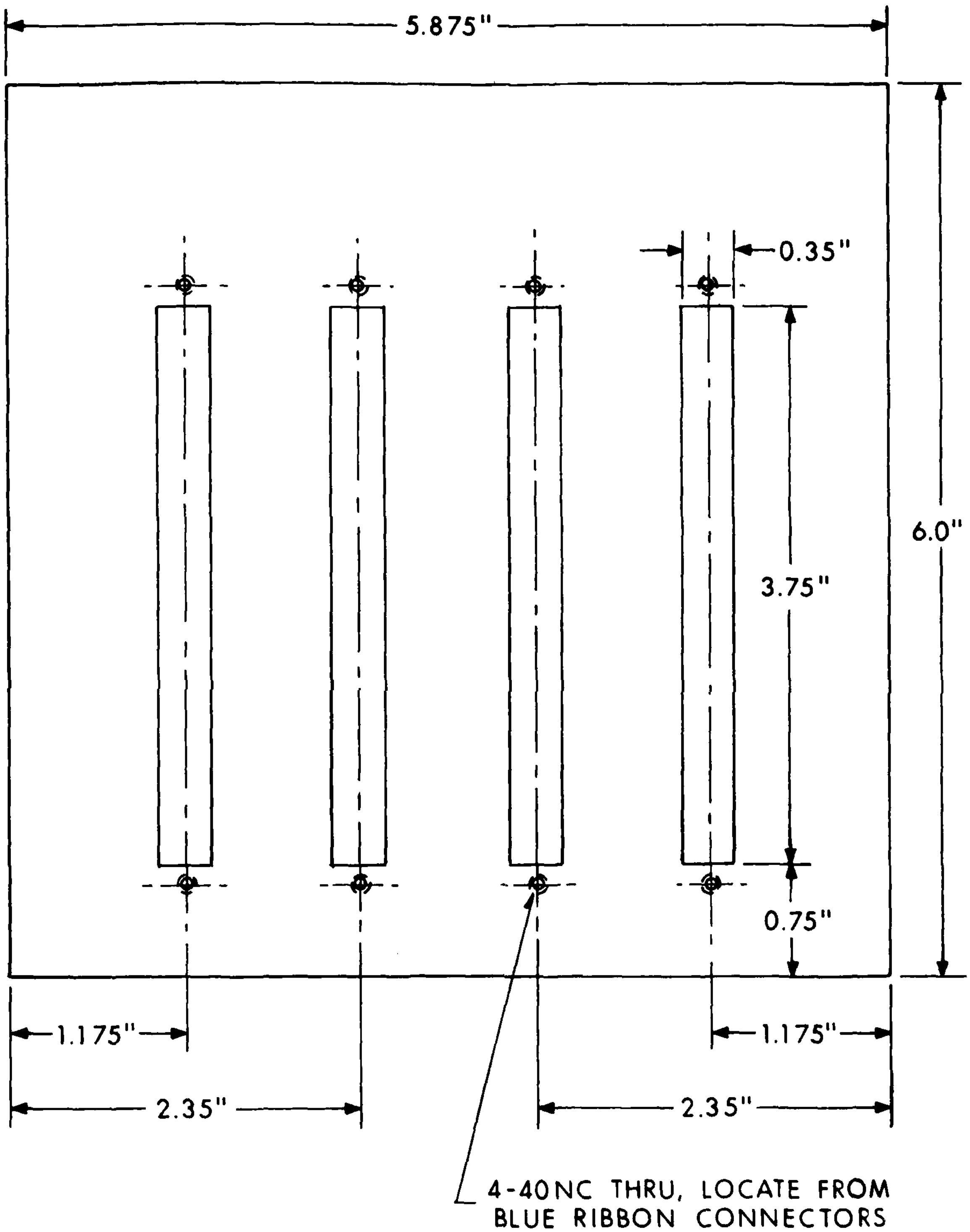


Figure 13. BLIS Slave Plug Panel

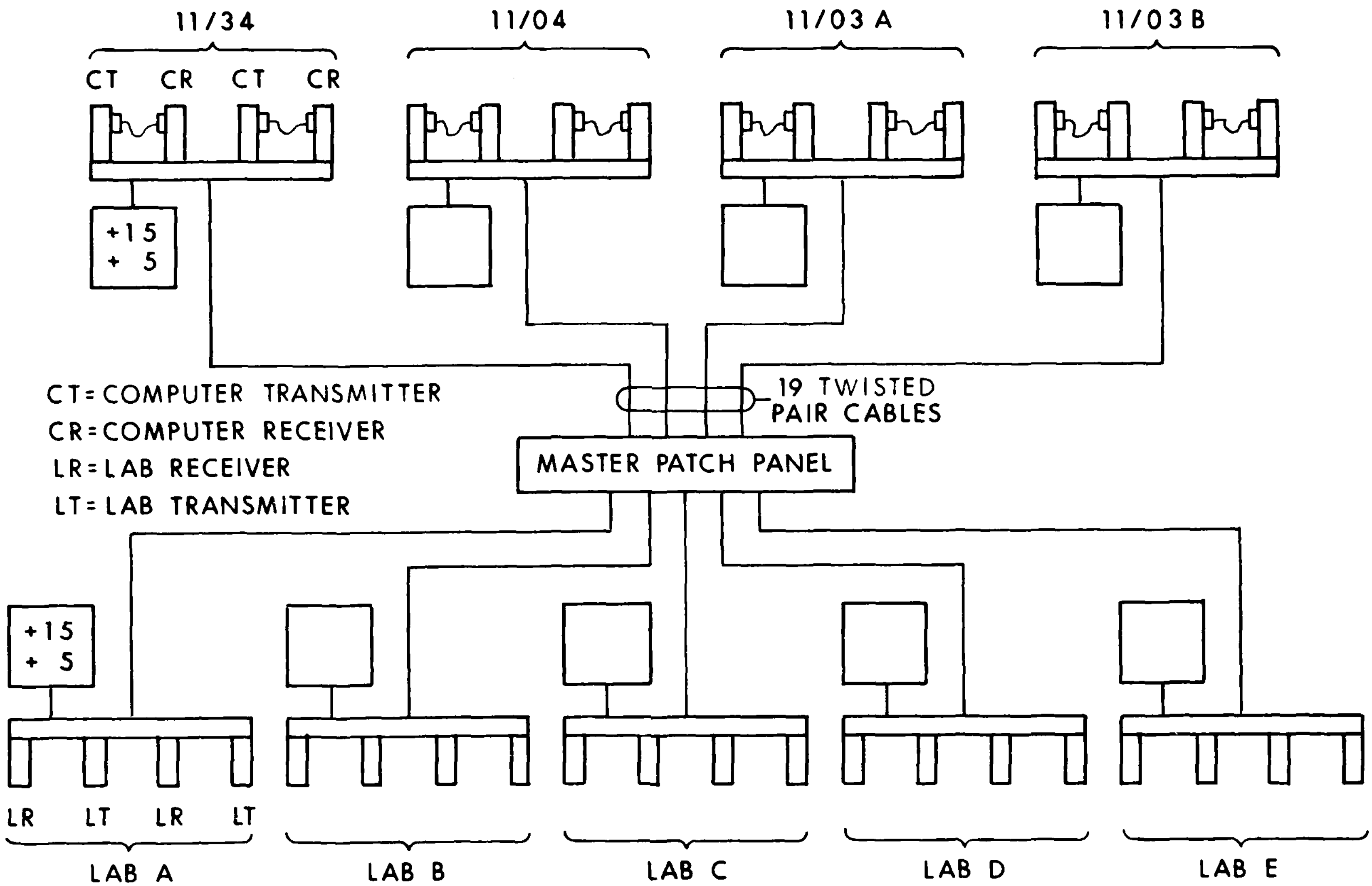


Figure 14. BLIS Overall System Schematic

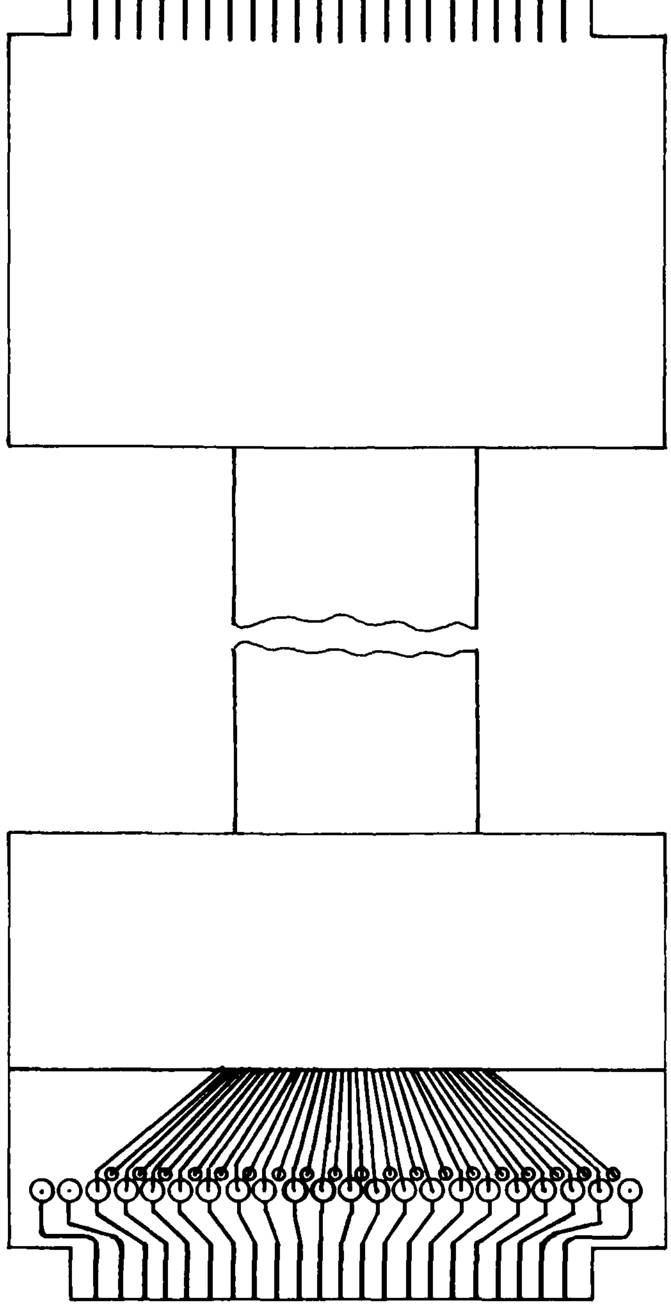


Figure 15. BLIS Patch Cable

Legend and Notes:

* The ground pins of the DR11-C are all connected to the ground plane of the computer end driver and receiver boards. The numbered edge connector pins 1, 2, and 22 are used for power to the boards. The numbered pins from 3 to 21 are used as current return lines for the driver boards and are tied to the common ground plane.

\# Ground signals are not carried from transmitters to receivers. The current loop formed by the driver current and the return path is terminated in the receiver board in the light emitting diode section of the opto-isolators. In this way, ground potentials at the two ends of the transmission line may have different levels, and not cause ground loop currents to flow.

** These signals are carried between the two boards of the computer end driver-receiver set by an eight pin jumper plug. This allows logically related signals to arrive at the laboratory on the same board.

CD Computer end driver board. This converts incoming TTL signals from the DR11-C to the current loop system and sends them out on the cables.

CR Computer end receiver board. This receives the current loop signals arriving at its edge fingers, and converts them to TTL at its Berg header for output to the DR-11C.

LD Laboratory end driver. This accepts TTL signals from laboratory instruments, and converts them to current loop format for transmission to a computer end receiver.

LR Laboratory end receiver. This accepts current loop format signals from the transmission line, and converts them to TTL format to control laboratory instruments.

As a final note about this system, it should be noted that this scheme provides very high immunity to electronic noise pickup. In order for the current loop portion to pick up noise, first the noise pulse must penetrate the cable outer shielding, the individual twisted pair's shielding, and still be able to generate a current pulse of 50 milliamps at a voltage of 0.6 volts for a time of five microseconds or longer. Such electronic noise sources as are able to perform this feat are usually known as lightning bolts, and occur more on golf courses than in laboratories. The TTL signals are more prone to noise pickup than the current loop ones, but due to the fact that a TTL signal in the "high" state is actively pulled up to the supply line by a transistor, and actively pulled down to the ground potential in the "low" state, it too has high noise immunity. A transmission system as is used in such systems as IEEE-488, uses instead, open collector TTL. This logic rather than clamping the TTL line to +5 with a transistor instead allows it to "float" up with the aid of a resistor of typically 1.2 to 1.8 K ohms. The result of this is high immunity to noise when TTL "low" is asserted, and almost none when TTL "high" is present. This deficiency has caused IEEE-488 controlled instrumentation in our laboratories to fail when large pulsed lasers fire, generating respectable quantities of non-shieldable electronic noise.

V. THE ANALOG INPUT SYSTEM

The next subsystem that will be discussed is the relatively low bandwidth analog input subsystem. The purpose of this system is to allow the system user to make voltage measurements of signals generated in the laboratory, at a rate of not more than 25,000 measurements per second. At the time of writing, limited use has been made of this system, in that the interfacing of more sophisticated instruments make the "brute force" method of clock-controlled periodic sampling and digitizing of analog waveforms almost unnecessary. Instead, this system has been used more as a means of monitoring slowly varying parameters of experimental setups such as laser power, pressures within systems, etc., for both data logging and malfunction detection. The system consists of three major components: the commercially available type AD-11 analog to digital converter circuit board, the type KW-11C programmable real-time clock circuit board, and the Analog Laboratory Interface System (ALIS) patching system. The details of the former device are described in References 2 and 4, and will not be repeated here, except for those necessary for coherence. As an additional note, the system here is only installed in the two system 11/03s at the time of writing. The system 11/04 is used in an application where no A/D converter unit is needed, being connected to a transient digitizer,[7] and the system 11/34 is equipped with a special type of converter system described elsewhere.[9]

The ALIS patch system provides the same function for analog signals that the BLIS patch system provides for binary TTL ones. As before, each laboratory is equipped with a custom-made slave panel, connected to identical rows of connectors in the master panel located in the same room as the computer system. The ALIS master patch panel is shown in Figure 16. The two configurations of the ALIS slave panels are shown in Figure 17. The connectors and cable used are Triax type, from Trompeter Electronics Company, part no. J72S-7 for the shielded patch jacks, part no. PL71-7 for the patch plugs, part no. TRC-50-2 for the 50 ohm triax cable, part no. LPLTR-50 for the looping plugs, and part no. TPT-50 for the terminating resistance plugs. The selection of triax cable was made over conventional coax cable since this made possible the transmission of differential signals over a cable with a grounded shield. As the A/D converters can accept single-ended, pseudo-differential or true differential signals (see Reference 10), this cable system provides the necessary versatility. The inputs to the A/D converters on the master patch panel for the 11/03 systems were wired in true differential mode. In this mode, there are eight analog inputs available, numbered 0 to 7 on the master patch panel. The jack numbered 8 is connected to the A/D converter external trigger input, and those numbered 9, and 10 are connected to the Schmitt trigger inputs available on the KW-11 clock board.

In order to protect the control circuits of the circuit boards, to provide a means to accept signals with differing logic relationships, and to provide the user with a convenient means to set up and adjust the various signals and levels of this subsystem, a control panel and circuit was made. The control panel is shown in Figure 18, and the control circuitry in Figure 19. The switches shown in Figures 18 and 19 are used to set the active levels of the control lines. S1 through S4 are used to control the inputs to the two system's external A/D converter's trigger inputs. S1 and S2 are for system 11/03A, while S3 and S4 are for system 11/03B. Opening both S1 and S2 prevents external triggers from reaching the ADC trigger of 11/03A. Closing

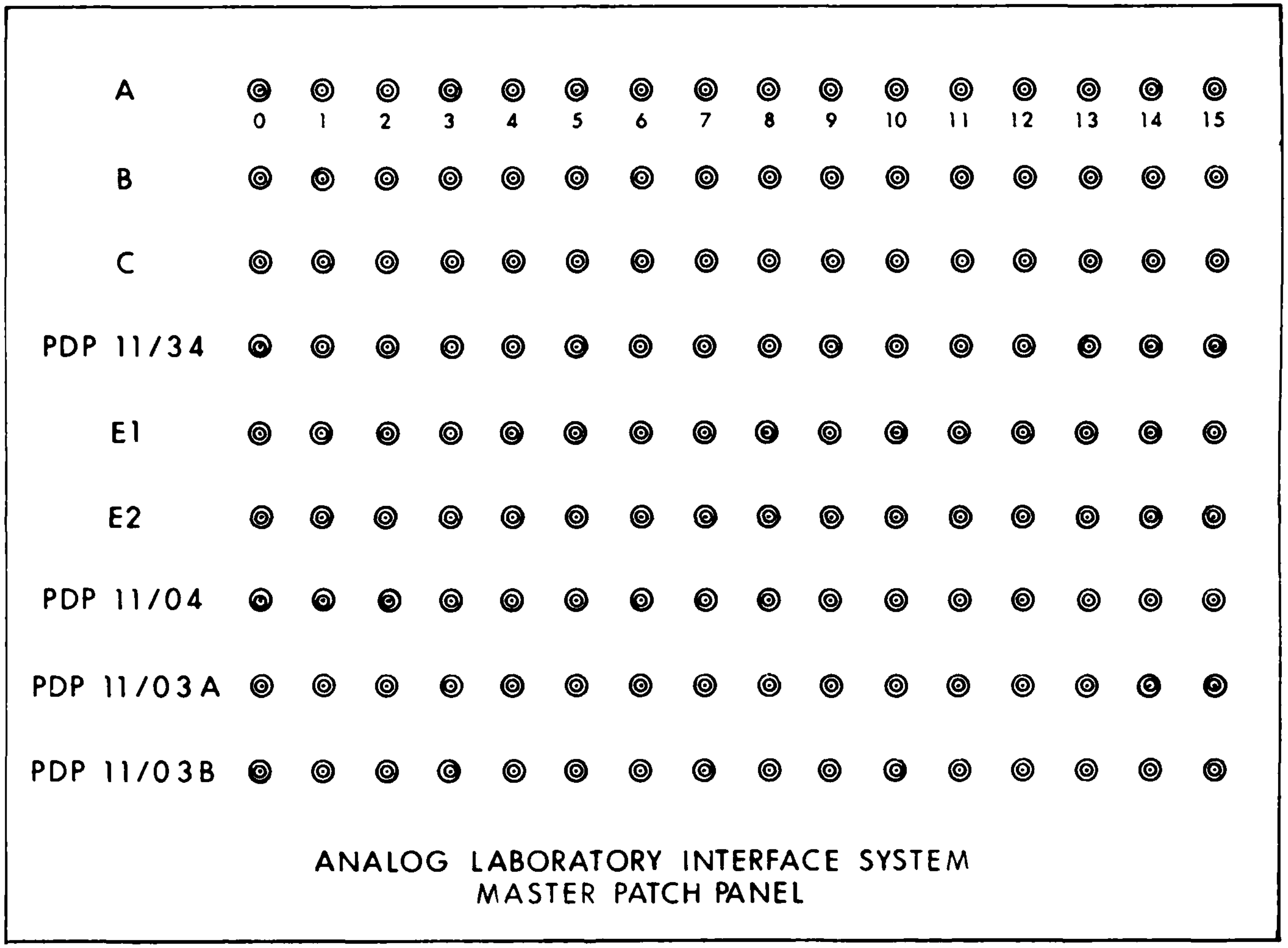


Figure 16. ALIS Master Patch Panel

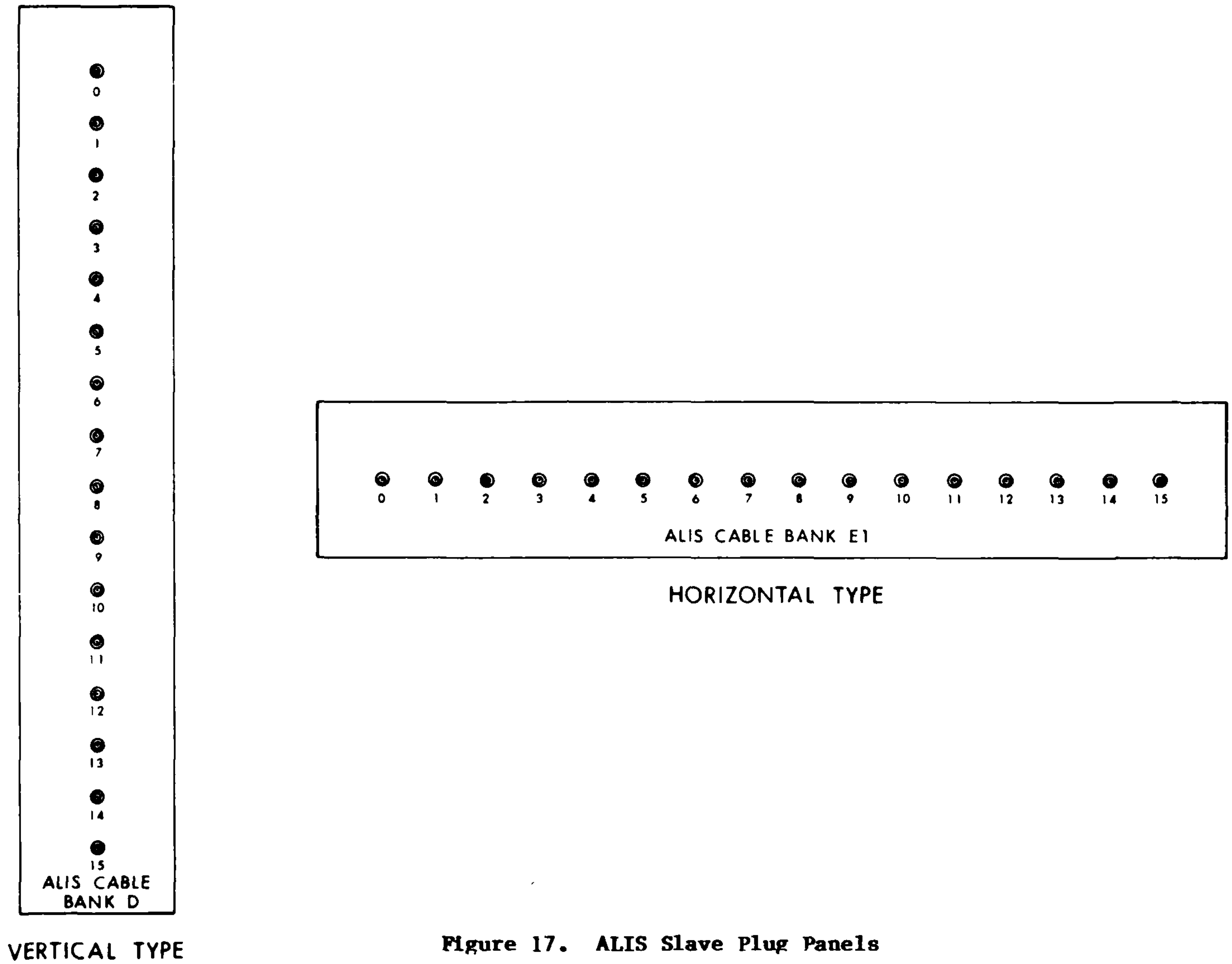


Figure 17. ALIS Slave Plug Panels

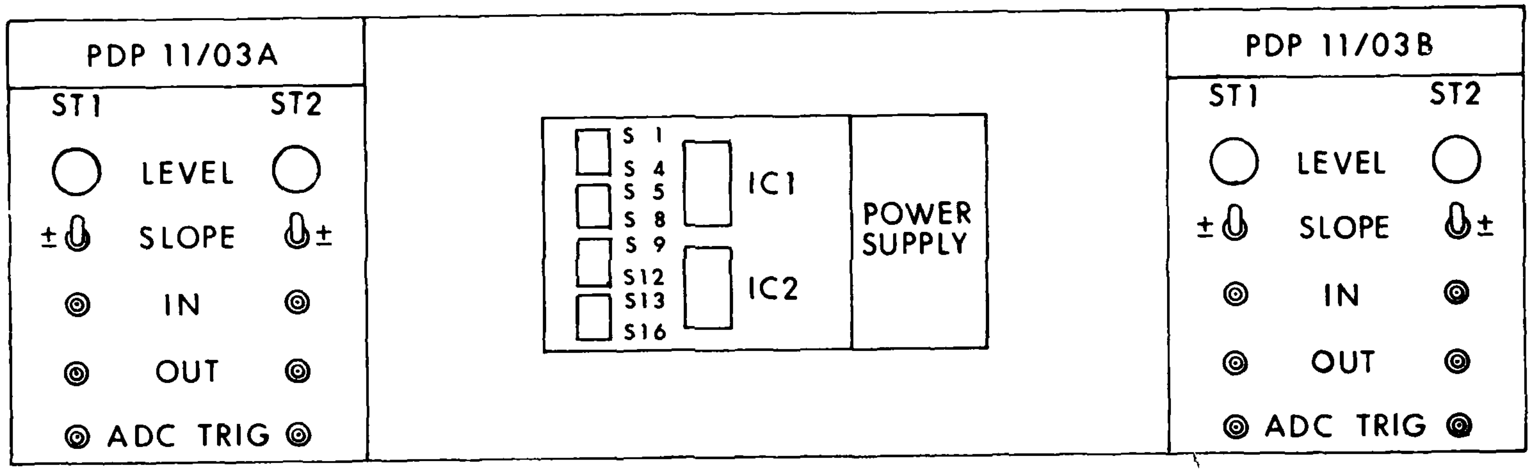


Figure 18. ALIS ADC Control Panel Layout

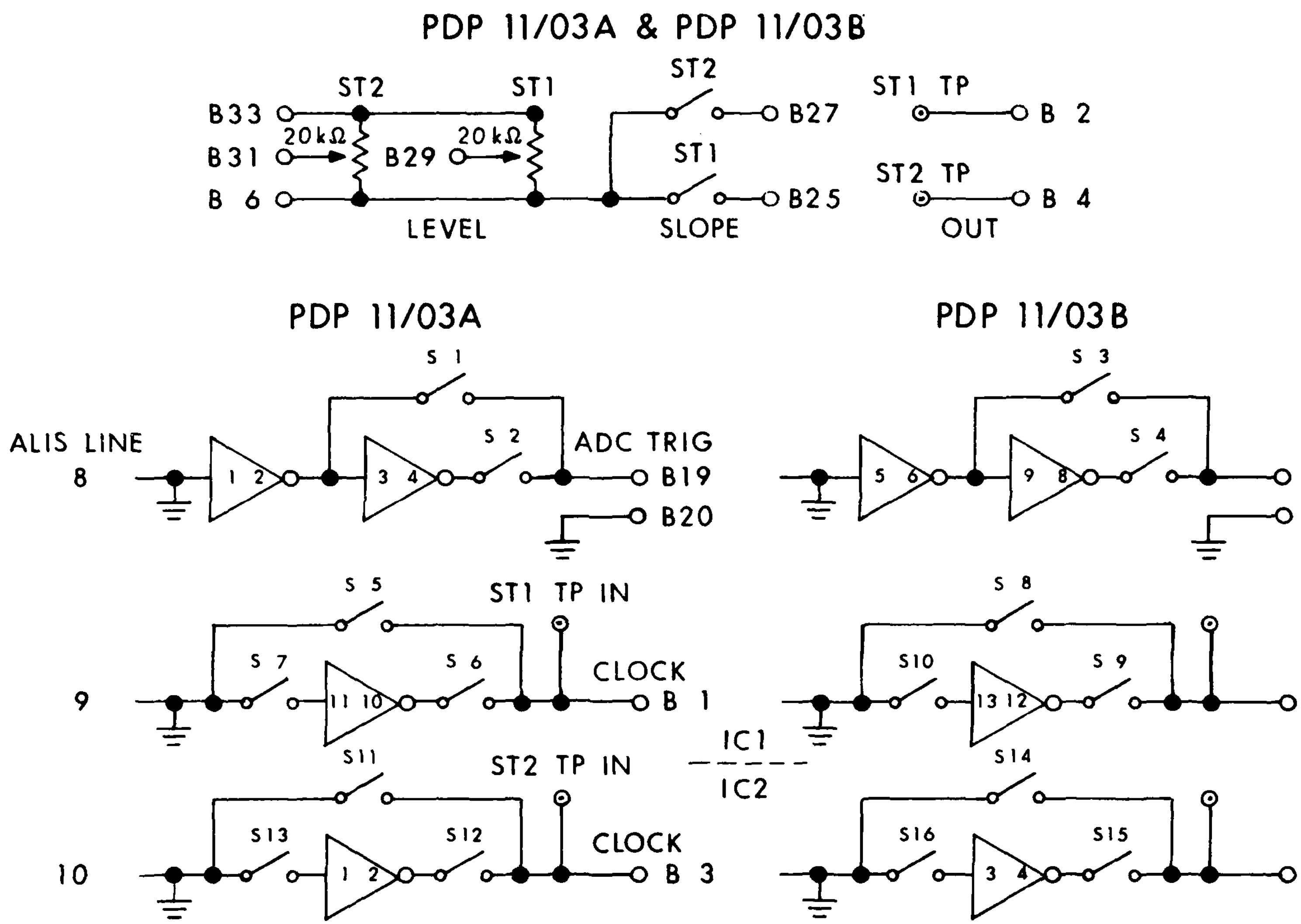


Figure 19. ALIS ADC Control Schematic

S2 only causes the input signal to go through the two Schmitt trigger inverters and arrive at the ADC in the same logic sense as it was received. In this mode, if enabled in the software, the 0 to 1 transition of the input trigger signal will cause an A/D conversion to begin. If S1 only is closed, then the input signal is inverted, and the conversion will commence on the 1 to 0 transition of the input trigger. Closing both switches leads to an indeterminate condition. S3 and S4 have identical functions as S1 and S2, but refer to system 11/03B. Switches S5 through S15 are used to control the inputs to the Schmitt trigger inputs of the clock board, and the following table summarizes the assignments:

Switch	Signal Controlled, System Assigned To
S1 - S2	ADC External trigger, PDP-11/03A
S3 - S4	ADC External trigger, PDP-11/03B
S5 - S7	Schmitt trigger 1, PDP-11/03A
S8 - S10	Schmitt trigger 1, PDP-11/03B
S11 - S12	Schmitt trigger 2, PDP-11/03A
S14 - S16	Schmitt trigger 2, PDP-11/03B

Referring to the schematic shown in Figure 19, it is seen that the Schmitt trigger control circuits are all identical, so that only one need be described. The PDP-11/03A Schmitt trigger 1 circuit will be used as the example. As the Schmitt triggers on the clock boards respond to analog voltage levels, it is desirable to be able to connect an external signal directly to these inputs. In the schematic, closing S5 with the other two open will accomplish this. In order to "clean up" TTL digital signals, a Schmitt trigger input digital inverter is used, and is put into circuit by opening S5, and closing S6 and S7. The remaining circuitry shown in the schematic provides the trigger levels and slopes that the Schmitt triggers will use. All pin numbers and other relevant data to construction is shown in the schematic.

VI. THE MACRO AND PASCAL A/D CONVERTER DRIVERS

In order to show an example of the writing of hardware device drivers in a high-level language, the device driver for a simple random sample grab mode of A/D conversion will be shown. The hardware is the A/D system described above, and the software will be written in both Pascal, and Macro-11 (assembly language). The function of the routines is to choose a single channel of the A/D converter and digitize the voltage level that appears there at the time of calling. The format of data is assumed to be two's complement (strappable option on A/D board), and the addresses of the A/D board to be 170400 and 170402, with an interrupt vector of 400. As before, all addresses are given in octal. The drivers do not make use of the interrupt capability of the unit, but rather query the "A/D done" bit in the CSR, bit 7. The multiplexer address is set in bits 8-11 of the same register, and bit 0 used to start a conversion. The device has a programmable gain of 1, 2, 4, or 8 and is selected by bits 2 and 3. The reader is referred to Reference 10 for details concerning the use of this feature. These drivers assume a gain of 1 for simplicity, but could easily be modified to offer other gains. The routines were made to be FORTRAN callable as subroutines, via the DEC standard calling protocol, and Pascal callable via Oregon Software's "Nonpascal" declaration of

procedures. The latter is not standard ISO Pascal, but rather a language extension by that vendor, as is the "Origin" declaration, which binds a variable to a specific absolute memory location. Finally, two other nonstandard Pascal constructs are used: bitwise "AND", and bitwise "OR". These two do the bit by bit logical operation of two numbers (integers), and leave the result in the indicated variable.

A brief discussion of the DEC calling protocol is in order. When a subroutine call is made in FORTRAN, such as:

CALL ADC1(ICHAN,IDATA)

the compiler assembles a data structure in memory that takes the form of a list. The first integer in that list is the number of arguments that the subroutine call had associated with it. The subsequent members of the list are the addresses of the variables associated with the arguments, NOT the arguments themselves. When the subroutine is called FORTRAN loads, register number 5 with the ADDRESS of the linkage list, and we say then that R5 points to the linkage list when the subroutine is entered. Pascal does the same convention, provided the procedure is declared "NONPASCAL" in the procedure header. In the call above, ICHAN is an integer from 0 to 7 that is the channel number of the ADC that is to be digitized. IDATA is an integer variable that will receive the data from the ADC, in numerical form. Since the ADC is a 12-bit type, the data will range from -4096 to +4095, which represents -10.0000 volts to +9.9951 volts at the input. If voltage is desired, a simple linear equation is used to convert these ADC "counts" to voltage. The driver below conforms to the DEC calling protocol for communication with the higher level language.

First, the driver in MACRO:

```
        .TITLE  ADC1                ;NAME OF ROUTINE
        .GLOBL  ADC1                ;FOR LINKER TO FIND IT BY
        .MCALL  .REGDEF             ;ALLOWS R0 INSTEAD OF %0, ETC.
        .REGDEF
        .ENABL  AMA                 ;TELLS ASSEMBLER TO USE ABSOLUTE MEMORY
                                    ;ADDRESSES WHENEVER POSSIBLE
        .ENABL  LSB                 ;ALLOWS LOCAL LABELS SUCH AS 1$, 2$, ETC.
        CSR = 170400                ;ADDRESS OF ADC'S CSR
        DATA = CSR + 2              ;ADDRESS OF ADC'S DATA BUFFER
        DONE = 200                  ;DONE BIT IS BIT 7, OR OCTAL 200
        GO = 1                      ;START ADC BIT IS BIT 1, OR OCTAL 1

ADC1:   TST     (R5)+               ;JUST POINTS R5 AT ADDRESS OF FIRST ARGUMENT
        MOVB    @(R5)+,1(CSR)       ;GETS THE ARGUMENT, AND PUTS IN UPPER
                                    ;BYTE OF CSR, WHICH IS THE MUX ADDRESS
        INC     R5                  ;TO POINT R5 AT ADDRESS OF NEXT
                                    ;ARGUMENT, SINCE AUTOINCREMENT ON BYTE
                                    ;INSTRUCTION ADDS 1, NOT 2 TO REGISTER
1$:     BIT     #DONE, CSR          ;MAKE SURE ADC IS NOT BUSY, I.E. DONE BIT = 1
        BEQ     1$                  ;AND LOOP UNTIL IT IS.
        BIS     #GO, CSR            ;NOW START ADC
```

```
2$:     BIT     #DONE, CSR          ;AND WAIT UNTIL DONE = 1 AGAIN
        BEQ     2$
        MOV     DATA, @(R5)         ;NOW MOVE DATA INTO SECOND ARGUMENT
        RTS     PC                  ;AND RETURN TO CALLING ROUTINE.
        .END    ADC1
```

And now the same driver in Pascal:

```
PROCEDURE ADC1(Chan : INTEGER; VAR Data : INTEGER);
EXTERNAL;          this would read "NONPASCAL" if it were to be called from
                   a FORTRAN program as a subroutine, instead of "EXTERNAL"

PROCEDURE ADC1;

  CONST
    Done = 200B;                    200B is only bit 7 in the integer = 1
    Start = 1B;                     only bit 0 is a "1"
    Shift = 256;                    multiply by this to shift 8 bits to left

  VAR
    CSR ORIGIN 170400B : INTEGER;
    DataBuffer ORIGIN 170402B : INTEGER;
    channel : INTEGER;

  BEGIN
    channel := channel * shift;           channel number into upper byte
    WHILE ((CSR AND Done) = 0) DO;        loop until done bit is "1"
    CSR := channel;                       load multiplexer address
    CSR := CSR OR Start;                  set the ADC "Start" bit
    WHILE ((CSR AND Done) = 0) DO;        loop until done bit = "1"
    Data := DataBuffer;                   get the converted data
  END;                                    return to calling routine
```

The two routines are identical in function. For this type of low bandwidth sampling, this type of driver works every bit as well as its assembly language counterpart, requires less care in writing, and is easier than its assembly language counterpart to understand. It should be noted that since the A/D converter takes 25 microseconds to convert the data to digital form, it, rather than the driver, is the limiting factor in the sampling rate. In this case, there is no loss in using the high level language rather than the assembly level one.

VII. THE MACRO AND PASCAL PLOTTING LIBRARIES

As mentioned earlier, the Tektronix 4014 and the various 4014 look-alike terminals in use may be driven for graphics by the Plot-10 software package for RT-11 available from Tektronix. This is a powerful package, and as a result, a large and complicated one. Whenever the system is used, it adds typically 8 Kbytes to the size of the program. This is often enough to force the use of overlays, and add to the complexity of organizing and writing the program. In addition, the package for RT-11 only allows a single set of hardware addresses on the bus to be used for the single terminal that will

receive the graphics. The reason for this is that the operating system intercepts certain characters being sent through the operating system to the terminal, and substitutes others. As an example, ESC is sent as a dollar sign, $. Since the 4014 requires the full ASCII character set to be sent, Tektronix uses its own terminal driver that expects the terminal to always have the same set of hardware addresses. In order to get around both of these problems, and provide the user with a compact albeit primitive set of graphics calls, plotting libraries were written both in Pascal, and MACRO-11. Although either could be called from Pascal, for reasons of software maintenance, the high-level language routines are used whenever possible, although the MACRO routines use less memory space. The routines are divided into two classes: those that are for doing graphics on 4014 and look-alike terminals, and those that are for using various special features of the look-alikes, notably DEC VT-100 terminals with advanced video option, and retrofitted with SG-200 graphics circuit boards from Selanar Corporation. The latter boards are described in detail in Reference 11, and only a few notes about them are mentioned here. The resolution of the Selanar graphics in Tektronix mode is 1024 horizontal by 240 vertical. The 4096 x 4096 Tektronix coordinates are mapped into this area, and scaled for plotting. As this is done by a microprocessor, time is needed to do this. If the baud rate of the terminal is set to 9600, it is possible for the plotting instructions to come faster than the scaling and rasterization can be performed. If this happens, "spikes" occur on the plots. In order to prevent these spikes, delays are put into the code to slow the sending of graphics commands, or the baud is rate dropped to 4800. As the MACRO-11 routines are more extensive, they will be described first, and then the Pascal routines that equal them shown.

The first routine initializes the system, and takes into account that the hardware addresses of the system console terminal may have been changed with the software described above. This is done by querying a location in memory that the operating system uses to keep track of these addresses, and then setting variables in the routines to those addresses. The routines in macro and Pascal follow:

```
;***************************************************************************
;* ROUTINE SETTRM        THIS ROUTINE MUST BE THE FIRST CALL TO THE PLOTTING *
;*                       LIBRARY.  IT INITIALIZES THE VARIABLE "CSR" TO THE  *
;*                       CORRECT VALUE FOR THE CONSOLE ADDRESS IN USE.       *
;*                                                                           *
;*                       FORMAT:                                             *
;*                                      CALL SETTRM                          *
;***************************************************************************


        .TITLE  SETTRM
        .GLOBL  SETTRM,CSR
        .MCALL  .REGDEF
        .REGDEF
```

```
SETTRM: MOV     R0,S0           ;PRESERVE REGISTER CONTENTS
        MOV     @#54,R0         ;GET START OF RESIDENT MONITOR
        ADD     #304,R0         ;ADD OFFSET TO WORD CONTAINING CSR ADDRESS
                                ;FOR CONSOLE TERMINAL
        MOV     (R0),CSR        ;STORE IT IN LOCATION "CSR"
        MOV     S0,R0           ;RESTORE REGISTER
        RTS     PC              ;RETURN TO CALLING ROUTINE
CSR:    .WORD   0               ;STORAGE FOR THE CSR ADDRESS
S0:     .WORD   0
        .END    SETTRM


;*****************************************************************************
;* ROUTINE TRMOUT       THIS ROUTINE OUTPUTS THE CHARACTER HELD IN R0 TO    *
;*                      THE TERMINAL THAT IS BEING USED AS THE CONSOLE      *
;*                      DEVICE.  CALLED ONLY BY MACRO ROUTINES.             *
;*****************************************************************************

        .TITLE  TRMOUT
        .GLOBL  TRMOUT,CSR
        .MCALL  .REGDEF
        .REGDEF

TRMOUT: MOV     R1,S1           ;PRESERVE REGISTERS
        MOV     @#CSR,R1        ;GET ADDRESS OF DL-11 CSR
LOOP:   BIT     #200,4(R1)      ;SEE IF TRANSMITTER IS BUSY
        BEQ     LOOP            ;LOOP IF SO
        MOVB    R0,6(R1)        ;OUTPUT THE CHARACTER IN R0
LOOP1:  BIT     #200,4(R1)      ;WAIT UNTIL DONE SENDING OUT
        BEQ     LOOP1
        MOV     S1,R1           ;RESTORE REGISTER
        RTS     PC              ;RETURN
S1:     .WORD   0
        .END    TRMOUT


;*****************************************************************************
;* ROUTINE MOVDRW       THIS ROUTINE IS TO MOVE THE WRITING "PEN" TO A      *
;*                      LOCATION ON THE SCREEN AND THEN TO DRAW A STRAIGHT  *
;*                      LINE TO ANOTHER LOCATION.  AT LEAST 1 MOVE MUST BE  *
;*                      DONE IN ORDER TO PUT THE TERMINAL IN GRAPHICS MODE. *
;*                      FOLLOWING THIS, ANY NUMBER OF DRAWS MAY BE DONE.    *
;*                      THE FORMATS OF THE CALLS ARE:                       *
;*                                                                          *
;*                      CALL MOVE(IX,IY)                                    *
;*                              TO MOVE TO IX,IY WITHOUT DRAWING            *
;*                                                                          *
;*                      CALL DRAW(IX,IY)                                    *
;*                              TO DRAW TO IX,IY                            *
;*                                                                          *
;*                      0 <= X <= 4095,  0 <= Y <= 3120                     *
;*                      NO INTERNAL OUT OF RANGE CHECKS ARE MADE!!!         *
;*****************************************************************************
```

```
        .TITLE MOVDRW
        .GLOBL  MOVE,DRAW,TRMOUT
        .MCALL  .REGDEF
        .REGDEF

MOVE:   MOVB    #35,@#GS        ;SETUP TO SEND "GS" CHARACTER
        BR      NEXT
DRAW:   CLRB    @#GS            ;SET TO SEND NULL INSTEAD OF "GS"
NEXT:   CLRB    EXB             ;
        MOV     R0,S0           ;
        MOV     @2(R5),R0       ;GET IX COORDINATE
        ASL     R0              ;PUT HIGH 7 BITS OF X IN UPPER BYTE
        CLC                     ;SET CARRY BIT = 0 FOR ROTATE
        RORB    R0              ;PUT 7 LOW BITS BACK IN POSITION, BIT 8 = 0
        SEC                     ;SET CARRY BIT = 1 FOR ROTATE
        RORB    R0              ;SHIFT LOW 2 BITS OF X INTO "EXTRA BYTE"
        RORB    EXB
        RORB    R0
        RORB    EXB
        MOVB    R0,LOX          ;SEND REMAINING 5 BITS OF LOW 7 TO LOX
        SWAB    R0              ;NOW GET HIGH BITS OF X
        BISB    #40,R0          ;GET RID OF JUNK
        MOVB    R0,HIX          ;AND PUT IN HIX
        MOV     @4(R5),R0       ;NOW GET IY COORDINATE
        ASL     R0              ;REPEAT ABOVE PERFORMANCE, EXCEPT FOR
                                ;DIFFERENT FLAG BITS
        SEC
        RORB    R0
        SEC
        RORB    R0
        RORB    EXB
        RORB    R0
        RORB    EXB
        MOVB    R0,LOY
        SWAB    R0
        BISB    #40,R0
        MOVB    R0,HIY
        CLC
        .REPT   4               ;NOW GET THE "EXTRA BYTE" ASSEMBLED
        RORB    EXB             ;WITH THE CORRECT FLAG BITS
        .ENDR
        BISB    #140,EXB
        MOVB    GS,R0           ;AND OUTPUT THE STRING OF CHARACTERS
        JSR     PC,TRMOUT
        MOVB    HIY,R0
        JSR     PC,TRMOUT
        MOVB    EXB,R0
        JSR     PC,TRMOUT
        MOVB    LOY,R0
        JSR     PC,TRMOUT
        MOVB    HIX,R0
        JSR     PC,TRMOUT
        MOVB    LOX,R0
        JSR     PC,TRMOUT
```

```
        MOV     SO,RO           ;RESTORE REGISTERS
        RTS     PC
GS:     .BYTE   0               ;DATA AREA
HIY:    .BYTE   0
EXB:    .BYTE   0
LOY:    .BYTE   0
HIX:    .BYTE   0
LOX:    .BYTE   0
SO:     .WORD   0
        .END    MOVDRW
```

For the above routine, Reference 12 should be consulted in order to aid in understanding the protocol of sending the coordinates to the 4014 or 4014 look-alike terminals. This mode of sending is compatible with either the 1024 x 1024 screen addressing of the normal 4014, or with the 4096 x 4096 enhanced graphics addressing of the terminals. In this mode, the 12 bits of coordinate information for the high resolution mode, or the 10 bits for the low resolution mode are encoded into either five or four characters of eight bits each. In all of these characters, the highest three bits are used as "flag" bits, and each of the components of the X and Y coordinates has a unique set. The lowest two bits of X, and the lowest two bits of Y along with the flag bits form the so-called "extra" byte. Terminals that do not support the 4096 x 4096 addressing ignore the bytes with the extra byte flag bits. The next more significant five bits of X and Y are sent in the so-called "Low X", and "low Y" bytes, with their respective flag bits, and the most significant five bits of X and Y sent in the High X and High Y bytes similarly.

```
;***************************************************************************
;* ROUTINE ALFMOD       THIS ROUTINE TAKES THE TERMINAL OUT OF GRAPHICS  *
;*                      MODE, AND PLACES IT IN ALPHANUMERIC MODE.  THIS  *
;*                      IS NECESSARY PRIOR TO OUTPUTTING ANY NON GRAPHIC *
;*                      CHARACTERS TO THE TERMINAL.                      *
;*                                                                       *
;*                      CALL FORMAT: CALL ALFMOD                         *
;***************************************************************************


        .TITLE  ALFMOD
        .GLOBL  ALFMOD,TRMOUT
        .MCALL  .REGDEF
        .REGDEF
ALFMOD: MOV     RO,SO           ;PRESERVE REGISTERS
        MOV     #37, RO         ;SEND "US" CHARACTER
        JSR     PC, TRMOUT
        MOV     #12000,RO       ;SET UP FOR SHORT DELAY
LOOP:   DEC     RO
        BNE     LOOP
        MOV     SO,RO           ;RESTORE REGISTER
```

```
        RTS     PC
S0:     .WORD   0
        .END    ALFMOD

;***************************************************************************
;* ROUTINE LINOUT       THIS ROUTINE OUTPUTS A STRING OF CHARACTERS ON THE *
;*                      TERMINAL, AT THE POSITION THE WRITING "PEN" WAS    *
;*                      LEFT AT BY THE PREVIOUS MOVE OR DRAW COMMAND.  THE *
;*                      STRING MUST BE TERMINATED BY A NULL BYTE.          *
;*                      CALLING CONVENTIONS:                               *
;*                                                                         *
;*                      CALL LINOUT ("This is a string")                   *
;*                        or if CARRAY is an array of characters,          *
;*                      CALL LINOUT(CARRAY)                                *
;***************************************************************************
        .TITLE  LINOUT
        .GLOBL  LINOUT,TRMOUT
        .MCALL  .REGDEF
        .REGDEF
LINOUT: MOV     R0,S0           ;SAVE REGISTERS
        MOV     R1,S1
        MOV     2(R5),R1        ;GET ADDRESS OF STRING
LOOP:   MOVB    (R1)+,R0        ;PUT CHARACTER IN R0
        BEQ     DONE            ;DETECT NULL BYTE AS END OF STRING
        JSR     PC,TRMOUT       ;OTHERWISE OUTPUT IT
        BR      LOOP            ;AND CONTINUE WITH STRING
DONE:   MOV     S0,R0           ;RESTORE REGISTERS
        MOV     S1,R1
        RTS     PC
S0:     .WORD   0
S1:     .WORD   0
        .END    LINOUT

;****************************************************************************
;* ROUTINE ERASE                ERASES THE SCREEN OF THE TERMINAL           *
;*                              CALL FORMAT: CALL ERASE                     *
;****************************************************************************

        .TITLE  ERASE
        .GLOBL  ERASE,TRMOUT,DELAY
        .MCALL  .REGDEF
        .REGDEF
ERASE:  MOV     R0,S0           ;SAVE REGISTER
        JSR     PC,DELAY        ;GIVE TERMINAL TIME TO SETTLE
        MOV     #33,R0          ;OUTPUT "ESC" CHARACTER
        JSR     PC,TRMOUT
        JSR     PC,DELAY        ;ALLOW TO SETTLE
        MOV     #14,R0          ;OUTPUT "FF" CHARACTER TO TERMINAL
        JSR     PC, TRMOUT
        JSR     PC, DELAY
        MOV     S0,R0           ;RESTORE REGISTER
        RTS     PC
S0:     .WORD   0
        .END    ERASE
```

```
;***************************************************************************
;* ROUTINE DELAY                PROVIDES A SHORT DELAY TO ALLOW TERMINALS TO*
;*                              SETTLE AND COMPLETE PRIOR OPERATION BEFORE  *
;*                              STARTING THE NEXT.                          *
;*                              CALLING FORMAT: CALL DELAY                  *
;***************************************************************************

        .TITLE  DELAY
        .GLOBL  DELAY
        .MCALL  .REGDEF
        .REGDEF
DELAY:  MOV     R0,S0           ;PRESERVE REGISTER
        MOV     #20000,R0       ;SET UP DELAY
LOOP:   DEC     R0
        BNE     LOOP
        RTS     PC
        MOV     S0,R0           ;RESTORE REGISTER
S0:     .WORD   0
        .END    DELAY

;***************************************************************************
;* ROUTINE COPY         INITIATES A SCREEN DUMP SMEAR COPY IF A HARDCOPY    *
;*                      DEVICE IS CONNECTED TO THE TERMINAL.  CALL FORMAT:  *
;*                              CALL COPY                                   *
;***************************************************************************

        .TITLE  COPY
        .GLOBL  COPY,TRMOUT
        .MCALL  .REGDEF
        .REGDEF
COPY:   MOV     R0,S0           ;SAVE REGISTER
        MOV     #33,R0          ;OUTPUT "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     #27,R0          ;OUTPUT "ETB" CHARACTER TO TERMINAL
        JSR     PC, TRMOUT
        MOV     S0,R0           ;RESTORE REGISTER
        RTS     PC
S0:     .WORD   0
        .END    COPY

;***************************************************************************
;* ROUTINE BELL         RINGS THE TERMINAL'S BELL                           *
;*                              CALL COPY                                   *
;***************************************************************************

        .TITLE  BELL
        .GLOBL  BELL,TRMOUT
        .MCALL  .REGDEF
        .REGDEF
BELL:   MOV     R0,S0           ;SAVE REGISTER
        MOV     #7,R0           ;OUTPUT "BEL" CHARACTER
        JSR     PC,TRMOUT
        MOV     S0,R0           ;RESTORE REGISTER
        RTS     PC
```

```
SO:     .WORD   0
        .END    BELL

;**************************************************************************
;* ROUTINE CHRSIZ        CHANGES THE SIZE OF CHARACTERS ON A 4014 WITH THE  *
;*                       ENHANCED GRAPHICS FEATURE.  THE CALLING FORMAT IS: *
;*                                                                          *
;*                       CALL CHRSIZ(ISIZE)                                 *
;*                           WHERE:                                         *
;*                                                                          *
;*                               I = 1 FOR LARGEST SIZE                     *
;*                                   2 FOR NEXT SMALLER SIZE                *
;*                                   3 FOR NEXT SMALLER SIZE BELOW 2        *
;*                                   4 FOR SMALLEST CHARACTERS              *
;**************************************************************************

        .TITLE  CHRSIZ
        .GLOBL  CHRSIZ,TRMOUT
        .MCALL  .REGDEF
        .REGDEF
CHRSIZ: MOV     R0,S0           ;PRESERVE REGISTERS
        MOV     #33,R0          ;SEND OUT "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     @2(R5),R0       ;GET ISIZE
        DEC     R0              ;MAKE IT AN ADDEND
        BIS     #70,R0          ;SO THAT EITHER "8","9",":" OR ";" WILL
                                ;BE SENT DEPENDING ON ISIZE
        JSR     PC,TRMOUT       ;SEND IT OUT
        MOV     S0,R0           ;RESTORE REGISTER
        RTS     PC
S0:     .WORD   0
        .END    CHRSIZ

;**************************************************************************
;* ROUTINE SETVEC        THIS ROUTINE SETS THE WRITING BEAM INTENSITY, AND  *
;*                       THE HARDWARE GENERATED VECTOR TYPE FOR 4014 AND    *
;*                       4014 LOOK ALIKES WITH THE ENHANCED GRAPHICS FEATURE*
;*                       THE CALLING FORMAT:  CALL SETVEC(I,J)              *
;*                                                                          *
;*                       WHERE:                                             *
;*                               I = 0 FOR NORMAL Z-AXIS                    *
;*                                   1 FOR DEFOCUSED Z-AXIS                 *
;*                                   2 FOR WRITE-THROUGH MODE ENABLE        *
;*                                                                          *
;*                               J = 0 FOR SOLID LINE VECTORS               *
;*                                   1 FOR DOTTED LINE VECTORS              *
;*                                   2 FOR DOT - DASH VECTORS               *
;*                                   3 FOR SHORT DASH VECTORS               *
;*                                   4 FOR LONG DASH VECTORS                *
;**************************************************************************

        .TITLE  SETVEC
        .GLOBL  SETVEC,TRMOUT
        .MCALL  .REGDEF
```

```
        .REGDEF
SETVEC: MOV     R0,S0           ;PRESERVE REGISTER
        MOV     #33,R0          ;SEND OUT "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     @2(R5),R0       ;GET "I"
        ASL     R0              ;MOVE THE NUMBER OVER 3 BITS TO LEFT
        ASL     R0
        ASL     R0
        BIS     @4(R5),R0       ;MOVE IN "J"
        BIS     #140,R0         ;MAKE IT THE CORRECT CHARACTER TO DO
                                ;THE FUNCTIONS
        JSR     PC,TRMOUT       ;AND SEND IT OUT
        MOV     S0,R0           ;RESTORE REGISTERS
S0:     .WORD   0
        .END

;***************************************************************************
;*   ROUTINE GRAFIN      THIS ROUTINE PUTS THE GRAPHIC INPUT MODE CURSORS    *
;*                       ON THE SCREEN FOR THE USER TO MOVE TO SOME POINT.   *
;*                       WHEN AT THE DESIRED POINT ANY CHARACTER IS TYPED    *
;*                       AND THE PROGRAM RETURNS THE X AND Y COORDINATES OF  *
;*                       THE CURSOR, ALONG WITH THE CHARACTER.  THE COORD-   *
;*                       COORDINATES ARE RETURNED IN 1024 X 1024 FORMAT.     *
;*                       THE FORM OF THE CALL IS:                            *
;*                                                                           *
;*                               CALL GRAFIN(IX,IY,ICHAR)                    *
;*                       WHERE:                                              *
;*                               IX = THE X COORDINATE OF THE CURSOR         *
;*                               IY = THE Y COORDINATE OF THE CURSOR         *
;*                            ICHAR = THE CHARACTER THAT THE USER TYPED      *
;***************************************************************************

        .TITLE  GRAFIN
        .GLOBL  GRAFIN,TRMOUT,DELAY
        .MCALL  .REGDEF,.TTYIN,.TTINR
        .REGDEF
GRAFIN: JSR     PC,DELAY        ;WAIT FOR TERMINAL TO SETTLE
        MOV     #33,R0          ;SEND OUT "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     #32,R0          ;SEND OUT "SUB" CHARACTER
        JSR     PC,TRMOUT
        BIS     #10000,@#44     ;SET I/O SYSTEM TO SPECIAL MODE
        .TTYIN                  ;GET CHARACTER INTO R0
        MOV     R0,@6(R5)       ;AND PUT IT INTO ICHAR
        .TTYIN                  ;GET HIGH 5 BITS OF X INTO R0
        BICB    #340,R0         ;CLEAR OUT GARBAGE
        MOVB    R0,C2           ;STORE IT AWAY IN HIGH BYTE OF X
        ASR     C1              ;MOVE 3 BITS TO RIGHT SINCE 10 BITS
        ASR     C1              ;ARE THE TOTAL NUMBER IN X
        ASR     C1
        .TTYIN                  ;GET LOW BYTE OF X, THE 5 LSB'S
        BICB    #340,R0         ;CLEAR OUT GARBAGE
        BISB    R0,C1           ;PUT IN LOW 5 BITS OF X
        .TTYIN                  ;GET HIGH BYTE OF Y
```

```
        BICB    #340,R0         ;CLEAR OUT GARBAGE
        MOVB    R0,C4           ;STORE IT AWAY IN HIGH BYTE OF Y
        ASR     C3              ;MOVE 3 BITS TO RIGHT SINCE 10 BITS
        ASR     C3              ;ARE THE TOTAL NUMBER IN Y
        ASR     C3
        .TTYIN                  ;GET LOW BYTE OF Y, THE 5 LSB'S
        BICB    #340,R0         ;CLEAR OUT GARBAGE
        BISB    R0,C3           ;PUT IN LOW 5 BITS OF Y
        .TTINR                  ;CLEAN OUT ANY TERMINATING CHARACTERS--
        .TTINR                  ;WITHOUT WAITING IF NONE AVAILABLE
        MOV     #15,R0          ;SEND "CR" CHARACTER
        JSR     PC,TRMOUT
        MOV     #12,R0          ;SEND "LF" CHARACTER.  CR-LF TURNS OFF
        JSR     PC,TRMOUT       ;GRAPHIC INPUT MODE
        BIC     #10000,@#44     ;TURN OFF SPECIAL MODE
        MOV     C1,@2(R5)       ;RETURN X
        MOV     C3,@4(R5)       ;RETURN Y
        RTS     PC
C1:     .BYTE   0
C2:     .BYTE   0
C3:     .BYTE   0
C4:     .BYTE   0
        .END    GRAFIN
```

Some additional routines were written in MACRO in order to make use of special features of the VT-100 terminals in use which were equipped with the enhanced video option. These terminals, as was previously mentioned, also had retrofit graphics boards which required special control sequences to switch between normal VT-100 mode, and TEKTRONIX look-alike mode. The routines to do these functions follow.

```
;***************************************************************************
;* ROUTINE TKMODE       THIS ROUTINE PUTS A VT-100 TERMINAL WITH SELANAR   *
;*                      RETROGRAPHICS INTO TEKTRONIX 4014 EMULATION MODE.  *
;*                      THE FORMAT OF THE CALL IS : CALL TKMODE            *
;***************************************************************************

        .TITLE  TKMODE
        .GLOBL  TKMODE,TRMOUT,DELAY
        .MCALL  .REGDEF
        .REGDEF
TKMODE: MOV     R0,S0           ;PRESERVE REGISTERS
        JSR     PC,DELAY        ;ALLOW TERMINAL TO SETTLE
        MOV     #33,R0          ;OUTPUT AN "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     #61,R0          ;SEND A "1" CHARACTER
        JSR     PC,TRMOUT
        JSR     PC,DELAY        ;ALLOW TERMINAL TO SETTLE
        MOV     S0,R0           ;RESTORE REGISTERS
        RTS     PC
S0:     .WORD   0
        .END    TKMODE
```

```
;****************************************************************************
;* ROUTINE VTMODE        THIS ROUTINE TAKES A VT100 TERMINAL WITH SELANAR    *
;*                       RETROGRAPHICS OUT OF TEKTRONIX 4014 MODE, AND PUTS  *
;*                       IT BACK IN VT100 MODE.  CALL FORMAT:  CALL VTMODE   *
;****************************************************************************

        .TITLE  VTMODE
        .GLOBL  VTMODE,TRMOUT,DELAY
        .MCALL  .REGDEF
        .REGDEF
VTMODE: MOV     R0,S0           ;PRESERVE REGISTERS
        JSR     PC,DELAY        ;ALLOW TERMINAL TO SETTLE
        MOV     #33,R0          ;OUTPUT AN "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     #176,R0         ;SEND A " " CHARACTER
        JSR     PC,TRMOUT
        JSR     PC,DELAY        ;ALLOW TERMINAL TO SETTLE
        MOV     #60,R0          ;SEND A "0" CHARACTER
        JSR     PC,TRMOUT
        JSR     PC,DELAY        ;ALLOW TERMINAL TO SETTLE
        MOV     #124,R0         ;SEND A "T" CHARACTER
        JSR     PC,TRMOUT
        JSR     PC,DELAY        ;ALLOW TERMINAL TO SETTLE
        MOV     S0,R0           ;RESTORE REGISTERS
        RTS     PC
S0:     .WORD   0
        .END    VTMODE

;****************************************************************************
;* ROUTINE VTPAGE        THIS ROUTINE CLEARS THE PAGE OF A VT-100 TERMINAL  *
;*                       OR VT-100 LOOK-ALIKE.  CALL:  CALL VTPAGE          *
;****************************************************************************

        .TITLE  VTPAGE
        .GLOBL  VTPAGE,TRMOUT,DELAY
        .MCALL  .REGDEF
        .REGDEF
VTPAGE: MOV     R0,S0           ;PRESERVE REGISTERS
        JSR     PC,DELAY        ;ALLOW TERMINAL TO SETTLE
        MOV     #33,R0          ;OUTPUT AN "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     #133,R0         ;SEND A "[" CHARACTER
        JSR     PC,TRMOUT
        JSR     PC,DELAY        ;ALLOW TERMINAL TO SETTLE
        MOV     #62,R0          ;SEND A "2" CHARACTER
        JSR     PC,TRMOUT
        JSR     PC,DELAY        ;ALLOW TERMINAL TO SETTLE
        MOV     #112,R0         ;SEND A "J" CHARACTER
        JSR     PC,TRMOUT
        JSR     PC,DELAY        ;ALLOW TERMINAL TO SETTLE
        MOV     S0,R0           ;RESTORE REGISTERS
        RTS     PC
S0:     .WORD   0
        .END    VTPAGE
```

```
;**************************************************************************
;* ROUTINE VMODES       THIS ROUTINE PUTS A VT-100 TERMINAL WITH ENHANCED   *
;*                      VIDEO OPTION INTO VARIOUS SPECIAL CHARACTER DISPLAY *
;*                      MODES.  THERE ARE 5 SEPARATE CALLS IN THIS ROUTINE: *
;*                                                                          *
;*                      CALL ALLOFF     TURNS OFF ALL SPECIAL ATTRIBUTES    *
;*                      CALL BOLD       MAKES ALL TYPE FULL INTENSITY       *
;*                      CALL UNDER      UNDERLINES ALL TYPED TEXT           *
;*                      CALL BLINK      CAUSES TYPE TO BLINK BRIGHT AND DIM *
;*                      CALL REVVID     CAUSES TYPE TO BE PRINTED IN        *
;*                                      REVERSE VIDEO- DARK TYPE ON LIGHT   *
;*                                      BACKGROUND                          *
;**************************************************************************

        .TITLE  VMODES
        .GLOBL  TRMOUT,ALLOFF,BOLD,UNDER,BLINK,REVVID
        .MCALL  .REGDEF
        .REGDEF

ALLOFF: MOV     #60,PS          ;SEND A "0" CHARACTER TO LOCATION "PS"
        BR      DOIT            ;GO TO "DOIT"
BOLD:   MOV     #61,PS          ;SEND A "1" CHARACTER TO LOCATION "PS"
        BR      DOIT
UNDER:  MOV     #64,PS          ;SEND A "4" CHARACTER TO LOCATION "PS"
        BR      DOIT
BLINK:  MOV     #65,PS          ;SEND A "5" CHARACTER TO LOCATION "PS"
        BR      DOIT
REVVID: MOV     #67,PS          ;SEND A "7" CHARACTER TO LOCATION "PS"
DOIT:   MOV     #33,R0          ;SEND OUT AN "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     #133,R0         ;SEND OUT A "[" CHARACTER
        JSR     PC,TRMOUT
        MOV     PS,R0           ;SEND OUT THE CHARACTER IN LOCATION "PS"
        JSR     PC,TRMOUT
        MOV     #155,R0         ;SEND OUT A "m" CHARACTER
        JSR     PC,TRMOUT
        MOV     S0,R0           ;RESTORE REGISTER
        RTS     PC
S0:     .WORD   0
PS:     .WORD   0
        .END    VMODES

;**************************************************************************
;* ROUTINE WIDER        THIS ROUTINE CAUSES CHARACTERS TO BE OUTPUT AS      *
;*                      DOUBLE WIDTH, SINGLE HEIGHT.  AFTER CALLING, THE    *
;*                      CHARACTERS MAY BE OUTPUT WITH ROUTINE LINOUT.       *
;*                      CALLING FORMAT : CALL WIDER                         *
;**************************************************************************

        .TITLE  WIDER
        .GLOBL  WIDER,TRMOUT
        .MCALL  .REGDEF
        .REGDEF
```

```
WIDER:  MOV     R0,S0           ;PRESERVE REGISTER
        MOV     #33,R0          ;SEND OUT "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     #43,R0          ;SEND OUT "#" CHARACTER
        JSR     PC,TRMOUT
        MOV     #66,R0          ;SEND OUT "6" CHARACTER
        JSR     PC,TRMOUT
        MOV     S0,R0           ;RESTORE REGISTER
        RTS     PC
S0:     .WORD   0
        .END    WIDER

;***************************************************************************
;* ROUTINE BIGOUT       THIS ROUTINE CAUSES A STRING TO BE OUTPUT AS       *
;*                      DOUBLE WIDTH, DOUBLE HEIGHT CHARACTERS ON THE VT-  *
;*                      100 TERMINAL WITH ENHANCED VIDEO OPTION.  THE      *
;*                      FORMS OF THE CALL ARE:                             *
;*                                                                         *
;*                      CALL BIGOUT('THIS IS THE STRING')                  *
;*                              OR                                         *
;*                      CALL BIGOUT(IARRY)                                 *
;*                              WHERE IARRY IS AN ARRAY OF BYTES CONTAINING*
;*                              THE STRING, TERMINATED BY A NULL BYTE.     *
;***************************************************************************

        .TITLE  BIGOUT
        .GLOBL  BIGOUT,TRMOUT,LINOUT
        .MCALL  .REGDEF
        .REGDEF
BIGOUT: MOV     R0,S0           ;PRESERVE REGISTERS
        MOV     #33,R0          ;SEND OUT "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     #43,R0          ;SEND OUT "#" CHARACTER
        JSR     PC,TRMOUT
        MOV     #63,R0          ;SEND OUT "3" CHARACTER
        JSR     PC,TRMOUT
        JSR     PC,LINOUT       ;SEND OUT TEXT STRING
        MOV     #12,R0          ;SEND OUT "LF" CHARACTER
        JSR     PC,TRMOUT
        MOV     #15,R0          ;SEND OUT "CR" CHARACTER
        JSR     PC,TRMOUT
        MOV     #33,R0          ;SEND OUT "ESC" CHARACTER
        JSR     PC,TRMOUT
        MOV     #43,R0          ;SEND OUT "#" CHARACTER
        JSR     PC,TRMOUT
        MOV     #64,R0          ;SEND OUT "4" CHARACTER
        JSR     PC,TRMOUT
        JSR     PC,LINOUT       ;SEND OUT TEXT STRING AGAIN
        MOV     #12,R0          ;SEND OUT "LF" CHARACTER
        JSR     PC,TRMOUT
        MOV     #15,R0          ;SEND OUT "CR" CHARACTER
        JSR     PC,TRMOUT
        MOV     #33,R0          ;SEND OUT "ESC" CHARACTER
        JSR     PC,TRMOUT
```

```
        MOV     #43,R0          ;SEND OUT "#" CHARACTER
        JSR     PC,TRMOUT
        MOV     #65,R0          ;SEND OUT "5" CHARACTER
        JSR     PC,TRMOUT
        MOV     S0,R0           ;RESTORE REGISTER
        RTS     PC
S0:     .WORD   0
        .END    BIGOUT
```

At the time of writing, not all of the above mentioned functions are implemented in Pascal. The primary reason it is possible to implement this type of routine under RT-11 is that the Pascal compiler uses its own input-output routines to the system, rather than the standard RT-11 system calls. In this way, characters that would normally be intercepted by the operating system prior to transmission to the terminal are sent regardless. The equivalents of the routines above that have been implemented in this way are move, draw, alfmod, settrm, vtmode, tkmode, erase, vtpage, grafin, and linout. Some of the routines are combined into composite functions and will be explained in the comments in the code. The procedures for the Pascal graphics package follow. Reference 13 is suggested in order to aid in the explanation of any language elements that may depart from the ISO standard Pascal language.

```
*****************************************************************************
     This routine clears the VT-100 screen, sets the terminal to TEKTRONIX
4014 mode, erases the 4014 mode screen, and sets the 4014 character set to
the largest size.  The routine works equally well on 4014 terminals, and 4014
look-alike terminals.  This routine does the work of settrm, tkmode, vtpage,
and erase in a single call.
*****************************************************************************


Procedure Initt;
   External;


Procedure Initt;
  VAR
    I,J : Integer;
  BEGIN
    Write(chr(33B),'$',chr(33B),'2',chr(33B),'[2J',chr(35B));
    Write(chr(33B),chr(14B),chr(33B),'8');
    For J := 1 to 3 Do
      Begin
        I := 1;
        While I < 32765 Do I := I + 1;
      End;
  End;    Procedure Initt
```

```
*************************************************************************
     This routine does the function of routine movdrw in the MACRO
package.  The variable "LastLowX" is used by the routine to cause the
writing beam to make a visible line on a draw, after a move.  The boolean
variable "draw" is made "true" to draw, and "false" to move.  Xpos and
ypos are the x and y coordinates for the move or draw.
*************************************************************************

Procedure MoveDraw(VAR LastLowX : Integer;
                   xpos,ypos : integer;
                   draw : Boolean);
  External;


Procedure Movedraw;
   VAR
     hix,lox,hiy,loy,exb : Integer;

   BEGIN
     exb := (ypos MOD 4) * 4 + (xpos MOD 4) + 140B;
     loy := 140B + (37B AND (ypos DIV 4));
     hiy := 40B + (37B AND (ypos DIV 128));
     lox := 100B + (37B AND (xpos DIV 4));
     hix := 40B + (37B AND (xpos DIV 128));
     write(chr(35B));
     if draw then write(chr(LastLowX));
     write(chr(hiy),chr(exb),chr(loy),chr(hix),chr(lox));
     LastLowX := lox;
   END;   Procedure MoveDraw


 *************************************************************************
     The following four procedures are the Pascal equivalents of the MACRO
routines VTMODE, ALFMOD, ERASE, and GRAFIN, respectively.
*************************************************************************

Procedure GoToVT100;
   BEGIN
     write(chr(33B),' OT');
   END;   GoToVT100

Procedure GoToAlpha;
   BEGIN
     write(chr(37B));
   END;   GoToAlpha

Procedure NewPage;
   VAR
     I,J : Integer;

   BEGIN
     write(chr(33B), chr(14B));
     For J := 1 to 3 Do
       BEGIN
```

```
      I := 1;
      While I < 32767 Do I := I + 1;
      END;
   END;   NewPage

Procedure GetScreenPoint(VAR X,Y : Integer; VAR InputCharacter : CHAR);
   VAR
     c0,c1,c2,c3,c4 : CHAR;

   BEGIN
     write(chr(33B),chr(32B));
     read(c0,c1,c2,c3,c4);
     readln;
     writeln;
     X := 4 * (32 * (37B AND ord(c1)) + (37B AND ord(c2)));
     Y := 4 * (32 * (37B AND ord(c3)) + (37B AND ord(c4)));
     InputCharacter := c0;
   END;   GetScreenPoint
```

It is easy to see that no routine such as linout is necessary, and also that the other routines not shown here are extremely simple to translate into Pascal. As a final note, when these MACRO and Pascal routines become object modules, they are generally incorporated into library files in the system, so that the library may be included as input to the system link program. The linker then extracts only those modules from the library that are actually needed. In this way, ease of use for the user is insured.

REFERENCES

1. Microcomputer and Memories Handbook, The Digital Equipment Corp., Maynard, MA, 1981.

2. Microcomputer Interfaces Handbook, The Digital Equipment Corp., Publication No. EB-20175-20/81, Maynard, MA, 1980.

3. PDP11/04/24/34a/44/70 Processor Handbook, The Digital Equipment Corp., Publication No. EB-19402-20/81, Maynard, MA, 1981.

4. Peripherals Handbook, The Digital Equipment Corp., Publication No. EB-18293-20/80, Maynard, MA, 1981.

5. Terminals and Communications Handbook, The Digital Equipment Corp., Publication No. EB-20752-20, Maynard, MA, 1981-1982.

6. DeWilde, M.A., "A Versatile Data Acquisition and Control System For Raman and Other Forms of Optical Spectroscopy," ARBRL-TR-in progress.

7. DeWilde, M.A., and Weaver, C.E., "An Automated System for the Control of, and Data Acquisition from Multiphoton Ionization and Fluorescence Lifetime Measurements," ARBRL-TR-in progress.

8. RT-11 Version 4 Documentation Kit, The Digital Equipment Corp. Publication No. 0J013-GZ, Maynard, MA, 1982.

9. DeWilde, M.A., and Weaver, C.E., "A Multi-Mode Subsystem for Moderate Bandwidth Analog to Digital Conversion of Laboratory Data," ARBRL-TR-in progress.

10. LSI-11 Analog System User's Guide, The Digital Equipment Corp., Publication No. EK-AXV11-UG-002, Maynard, MA, 1982.

11. Selanar Graphics 200 Manual, Selanar Corp., Santa Clara, CA, 1982.

12. 4014 and 4014-1 Computer Display Terminal User's Manual, Tektronix Corp., Publication Number 070-1647-00, Beaverton, OR, 1974.

13. Pascal-2 User's Manual for RT-11, Oregon Software, Inc., Portland, OR, 1982.

DISTRIBUTION LIST

No of Copies	Organization
12	Administrator Defense Technical Info Center ATTN: DTIC-DDA Cameron Station Alexandria, VA 22314
1	HQ DA DAMA-ART-M Washington, DC 20310
1	Commander US Army Materiel Cmd. ATTN: AMCDRA-ST 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander Armament R&D Center USA AMCCOM ATTN: SMCAR-TDC Dover, NJ 07801-5001
2	Commander Armament R&D Center USA AMCCOM ATTN: SMCAR-TSS Dover, NJ 07801-5001
1	Commander US Army Armament, Munitions and Chemical Cmd ATTN: SMCAR-ESP-L Rock Island, IL 61299
1	Director Benet Weapon Laboratory Armament R&D Center USA AMCCOM ATTN: SMCAR-LCB-TL Watervliet, NY 12189
1	Commander USA Aviation Rsch and Development Cmd ATTN: AMSAV-E 4300 Goodfellow Blvd. St. Louis, MO 63120

No. of Copies	Organization
1	Director USA Air Mobility Rsch and Development Lab. Ames Research Center Moffett Field, CA 94035
4	Commander USA Research Officer ATTN: R. Ghirardelli D. Mann R. Singleton R. Shaw Research Triangle Park NC 27709
1	Commander U.S. Army Communications - Electronics Command ATTN: AMSEL-ED Fort Monmouth, NJ 07703
1	Commander U.S. Army Communications - Electronics Command (CECOM) CECOM R&D Technical Library ATTN: AMSEL-IM-L B 2700 Fort Monmouth, NJ 07703-5000
2	Commander USA AMCCOM ATTN: DRSMC-LCA-G D.S. Downs J.A. Lannon Dover, NJ 07801
1	Commander USA AMCCOM ATTN: DRSMC-LC, L. Harris Dover, NJ 07801
1	Commander USA AMCCOM ATTN: DRSMC-SCA-T L. Stiefel Dover, NJ 07801

DISTRIBUTION LIST

No of Copies	Organization
1	Commander USA Missile Command Research, Development, Engineering Center ATTN: AMSMI-RD Redstone Arsenal, AL 35898
1	Commander USA Missile & Space Command Intelligence Center ATTN: AIAMS-YDL Redstone Arsenal, AL 35898
2	Commander USA Missile Command ATTN: DRSMI-RK, D.J. Ifshin Redstone Arsenal, AL 35898
1	Commander USA Tank Automotive Cmd ATTN: AMSTA-TSL Warren, MI 48397-5000
1	Director USA TRADOC Systems Analysis ATTN: ATAA-SL WSMR, NM 88002
2	Commandant USA Infantry School ATTN: ATSH-CD-CSO-OR Fort Benning, GA 31905
1	Commander USA Development and Employment Agency ATTN: MODE-TED-SAB Fort Lewis, WA 98433
1	Office of Naval Rsch Department of the Navy ATTN: R.S. Miller, Code 432 800 N. Quincy Street Arlington, VA 22217

No. of Copies	Organization
1	Navy Strategic Systems Project Office ATTN: R.D. Kinert, SP 2721 Washington, DC 20376
1	Commander Naval Air Systems Cmd ATTN: J. Ramnarace, AIR-54111C Washington, DC 20360
3	Commander Naval Ordnance Station ATTN: C. Irish S. Mitchell P.L. Stang, Code 515 Indian Head, MD 20640
1	Commander Naval Surface Weapons Center ATTN: J.L. East, Jr., G-20 Dahlgren, VA 22448
2	Commander Naval Surface Weapons Center ATTN: R. Bernecker, R-13 G.B. Wilmot, R-16 Silver Spring, MD 20910
4	Commander Naval Weapons Center ATTN: R.L. Derr, Code 389 China Lake, CA 93555
2	Commander Naval Weapons Center ATTN: Code 3891, T. Boggs K.J. Graham China Lake, CA 93555
5	Commander Naval Research Lab, Code 6110 ATTN: L. Harvey J. McDonald E. Oran J. Shnur R.J. Doyle, Washington, DC 20375

DISTRIBUTION LIST

No of Copies	Organization
1	Commanding Officer Naval Underwater System Center Weapons Dept. ATTN: R.S. Lazar/Code 36301 Newport, RI 02840
1	Superintendent Naval Postgraduate School Dept. of Aeronautics ATTN: D.W. Netzer Monterey, CA 93940
6	AFRPL (DRSC) ATTN: R. Geisler D. George B. Goshgarian J. Levine W. Roe D. Weaver Edwards AFB, CA 93523
1	Air Force Armament Laboratory ATTN: AFATL/DLODL ATTN: O.K. Heiney Eglin AFB, FL 32542-5000
2	AFOSR ATTN: L.H. Caveny J.M. Tishkoff Bolling Air Force Base Washington, DC 20332
1	AFWL/SUL Kirtland AFB, NM 87117
1	NASA Langley Research Center ATTN: G.B. Northam/MS 168 Hampton, VA 23365
4	National Bureau of Standards ATTN: J. Hastie M. Jacox T. Kashiwagi H. Semerjian US Dept. of Commerce Washington, DC 20234

No. of Copies	Organization
1	Aerojet Solid Propulsion Company ATTN: P. Micheli Sacramento, CA 95813
1	Applied Combustion Technology, Inc. ATTN: A.M. Varney P.O. Box 17885 Orlando, FL 32860
2	Atlantic Research Corp. ATTN: M.K. King 5390 Cherokee Avenue Alexandria, VA 22314
1	Atlantic Research Corp. ATTN: R.H.W. Waesche 7511 Wellington Road Gainesville, VA 22065
1	AVCO Everett Research laboratory Division ATTN: D. Stickler 2385 Revere Beach Parkway Everett, MA 02149
1	Battelle Memorial Institute Tactical Technology Center ATTN: J. Huggins 505 King Avenue Columbus, OH 43201
2	Exxon Research & Engineering Company ATTN: A. Dean M. Chou P.O. Box 45 Linden, NJ 07036
1	Ford Aerospace and Communications Corp. DIVAD Division Div. Hq., Irvine ATTN: D. Williams Main Street & Ford Road Newport Beach, CA 92663

DISTRIBUTION LIST

No of Copies	Organization
1	General Electric Armament & Electrical Systems ATTN: M.J. Bulman Lakeside Avenue Burlington, VT 05402
1	General Electric Co. ATTN: M. Lapp Schenectady, NY 12301
1	General Electric Ordnance Systems ATTN: J. Mandzy 100 Plastics Avenue Pittsfield, MA 01203
1	General Motors Rsch Lab Physics Department ATTN: R. Teets Warren, MI 48090
3	Hercules, Inc. Alleghany Ballistics Lab. ATTN: R.R. Miller P.O. Box 210 Cumberland, MD 21501
3	Hercules, Inc. Bacchus Works ATTN: K.P. McCarty P.O. Box 98 Magna, UT 84044
1	Hercules, Inc. AFATL/DLDL ATTN: R.L. Simmons Eglin AFB, FL 32542
1	Honeywell, Inc. Defense Systems Div. ATTN: D.E. Broden/ MS MN50-2000 600 2nd Street NE Hopkins, MN 55343
10	Central Intelligence Agency Office of Central Reference Dissemination Branch Room GE-47 HQS Washington, D.C. 20502

No. of Copies	Organization
1	IBM Corporation ATTN: A.C. Tam Research Division 5600 Cottle Road San Jose, CA 95193
1	Director Lawrence Livermore National Laboratory ATTN: C. Westbrook Livermore, CA 94550
1	Lockheed Missiles & Space Company ATTN: George Lo 3251 Hanover Street Dept. 52-35/B204/2 Palo Alto, CA 94304
1	Los Alamos National Lab ATTN: B. Nichols T7, MS-B284 P.O. Box 1663 Los Alamos, NM 87545
1	Olin Corporation Smokeless Powder Operations ATTN: R.L. Cook P.O. Box 222 St. Marks, FL 32355
1	Paul Gough Associates ATTN: P.S. Gough 1048 South Street Portsmouth, NH 03801
2	Princeton Combustion Research Labs, Inc. ATTN: M. Summerfield N.A. Messina 475 US Highway One Monmouth Junction, NJ 08852
1	Hughes Aircraft Company ATTN: T.E. Ward 8433 Fallbrook Avenue Canoga Park, CA 91303

DISTRIBUTION LIST

No of Copies	Organization	No. of Copies	Organization
1	Rockwell International Corporation Rocketdyne Division ATTN: J.E. Flanagan/HB02 6633 Canoga Avenue Canoga Park, CA 91304	1	University of Dayton Research Institute ATTN: D. Campbell AFRPL/PAP Stop 24 Edwards AFB, CA 93523
3	Sandia National Labs. Combustion Science Dept ATTN: R. Cattolica D. Stephenson P. Mattern Livermore, CA 94550	1	University of Florida Dept. of Chemistry ATTN: J. Winefordner Gainesville, FL 32611
1	Sandia National Labs. ATTN: M. Smooke Division 8353 Livermore, CA 94550	3	Georgia Inst. of Technology School of Aerospace Eng. ATTN: E. Price Atlanta, GA 30332
1	Science Applications, Inc. ATTN: R.B. Edelman 23146 Cumorah Crest Woodland Hills, CA 91364	2	Georgia Institute of Technology School of Aerospace Eng. ATTN: W.C. Strahle B.T. Zinn Atlanta, GA 30332
2	Univ. of California, Santa Barbara Quantum Institute ATTN: K. Schofield M. Steinberg Santa Barbara, CA 93106	1	University of Illinois Dept. of Mech. Eng. ATTN: H. Krier 144 MEB, 1206 W.Green Street Urbana, IL 61801
1	University of Southern California Dept. of Chemistry ATTN: S. Benson Los Angeles, CA 90007	1	Johns Hopkins Univ./APL Chemical Propulsion Information Agency ATTN: T.W. Christian Johns Hopkins Road Laurel, MD 20707
1	Case Western Reserve University Division of Aerospace Science ATTN: J. Tien Cleveland, OH 44135	1	University of Minnesota Dept. of Mechanical Eng. ATTN: E. Fletcher Minneapolis, MN 55455
1	Cornell University Department of Chemistry ATTN: E. Grant Baker Laboratory Ithaca, NY 14853	4	Pennsylvania State University Applied Research Lab. ATTN: G.M. Faeth K.k. Kuo H. Palmer M. Micci University Park, PA 16802

DISTRIBUTION LIST

No of Copies	Organization
1	Polytechnic Institute of NY ATTN: S. Lederman Route 110 Farmingdale, NY 11735
2	Princeton University Forrestal Campus Library ATTN: K. Brezinsky I. Glassman P.O. Box 710 Princeton, NJ 08540
1	Princeton University MAE Dept. ATTN: F.A. Williams Princeton, NJ 08544
1	Science Applications, Inc. ATTN: H.S. Pergament 1100 State Road, Bldg. N Princeton, NJ 08540
1	Space Sciences, Inc. ATTN: M. Farber Monrovia, CA 91016
4	SRI International ATTN: S. Barker D. Crosley D. Golden Tech. Lib 333 Ravenwood Avenue Menlo Park, CA 94025
1	Stevens Institute of Technology Davidson Laboratory ATTN: R. McAlevy, III Hoboken, NJ 07030
1	Teledyne McCormack-Selph ATTN: C. Leveritt 3601 Union Road Hollister, CA 95023
1	Thiokol Corporation Elkton Division ATTN: W.N. Brundige P.O. Box 241 Elkton, MD 21921
3	Thiokol Corporation Huntsville Division ATTN: D.A. Flanagan Huntsville, AL 35807
3	Thiokol Corporation Wasatch Division ATTN: J.A. Peterson P.O. Box 524 Brigham City, UT 84302
1	United Technologies ATTN: A.C. Eckbreth East Hartford, CT 06108
2	United Technologies Corp. ATTN: R.S. Brown R.O. McLaren P.O. Box 358 Sunnyvale, CA 94088
1	Universal Propulsion Company ATTN: H.J. McSpadden Black Canyon Stage 1 Box 1140 Phoenix, AX 85029
1	Veritay Technology, Inc. ATTN: E.B. Fisher P.O. Box 22 Bowmansville, NY 14026
1	Brigham Young Univ. Dept. of Chemical Eng. ATTN: M.W. Beckstead Provo, UT 84601
1	California Institue of Technology Jet Propulsion Lab. ATTN: MS 125/159 4800 Oak Grove Drive Pasadena, CA 91103

DISTRIBUTION LIST

No of Copies	Organization
1	California Institue of Technology ATTN: F.E.C. Culick/ MC 301-46 204 Karman Lab. Pasadena, CA 91125
1	Univ. of California, Berkeley Mechanical Engineering Department ATTN: J. Daily Berkeley, CA 94720
1	Univ. of California Los Alamos National Laboratory ATTN: T.D. Butler P.O. Box 1663, Mail Stop B216 Los Alamos, NM 87545
2	Purdue University School of Aeronautics and Astronautics ATTN: R. Glick J.R. Osborn Grissom Hall West Lafayette, IN 47907
3	Purdue University School of Mechanical Engineering ATTN: N.M. Laurendeau S.N.B. Murthy D. Sweeney TSPC Chaffee Hall West Lafayette, IN 47906
1	Rensselaer Polytechnic Institute Dept. of Chemical Engineering ATTN: A. Fontijn Troy, NY 12181

No. of Copies	Organization
2	Southwest Research Institute ATTN: R.E. White A.B. Wenzel 8500 Culebra Road San Antonio, TX 78228
1	Stanford University Dept. of Mechanical Engineering ATTN: R. Hanson Stanford, CA 93106
1	University of Texas Dept. of Chemistry ATTN: W. Gardiner Austin, TX 78712
1	University of Utah Dept. of Chemical Engineering ATTN: G. Flandro Salt Lake City, UT 84112
1	Virginia Polytechnic Institute & State University ATTN: J.A. Schetz Blackburg, VA 24061

Aberdeen Proving Ground

Dir, USAMSAA
ATTN: AMXSY-D
AMXSY-MP, H. Cohen
Cdr, USATECOM
ATTN: AMSTE-TO-F
Cdr, CRDC, AMCCOM
ATTN: SMCCR-RSP-A
SMCCR-MU
SMCCR-SPS-IL

USER EVALUATION SHEET/CHANGE OF ADDRESS

This Laboratory undertakes a continuing effort to improve the quality of the reports it publishes. Your comments/answers to the items/questions below will aid us in our efforts.

1. BRL Report Number____________________ Date of Report__________

2. Date Report Received______________________________

3. Does this report satisfy a need? (Comment on purpose, related project, or other area of interest for which the report will be used.)__________

4. How specifically, is the report being used? (Information source, design data, procedure, source of ideas, etc.)__________

5. Has the information in this report led to any quantitative savings as far as man-hours or dollars saved, operating costs avoided or efficiencies achieved, etc? If so, please elaborate.__________

6. General Comments. What do you think should be changed to improve future reports? (Indicate changes to organization, technical content, format, etc.)

CURRENT ADDRESS

Name

Organization

Address

City, State, Zip

7. If indicating a Change of Address or Address Correction, please provide the New or Correct Address in Block 6 above and the Old or Incorrect address below.

OLD ADDRESS

Name

Organization

Address

City, State, Zip

(Remove this sheet along the perforation, fold as indicated, staple or tape closed, and mail.)

FOLD HERE

Director
U.S. Army Ballistic Research Laboratory
ATTN: SLCBR-DD-T
Aberdeen Proving Ground, MD 21005-5066

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

OFFICIAL BUSINESS

PENALTY FOR PRIVATE USE, $300

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO 12062 WASHINGTON, DC

POSTAGE WILL BE PAID BY DEPARTMENT OF THE ARMY

Director
U.S. Army Ballistic Research Laboratory
ATTN: SLCBR-DD-T
Aberdeen Proving Ground, MD 21005-9989

FOLD HERE

END

12-86

DTIC